THREE ESSAYS ON PRODUCTION PLANNING IN SEMICONDUCTOR MANUFACTURING

By

BÜLENT ÇATAY

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1999

"Knowledge becomes wisdom only after it has been put to practical use."

(Anonymous)

This dissertation is dedicated to my parents, Mesut and Gülşen Çatay

ACKNOWLEDGEMENTS

I would like to express my appreciation of the members of my committee, Drs. S. Selcuk Erenguc, Asoo J. Vakharia, Daniel J. Conway, and Sherman X. Bai. I am deeply indebted to my advisors, Dr. Erenguc and Dr. Vakharia, for their insights, guidance and support throughout this research. I am also grateful to my teachers and professors, from elementary school to graduate school, who have sculpted my intellect throughout my academic life.

I owe the greatest dept to my parents for their sponsorship of my studies and my life in general, and for their continuous support through all ups and downs.

TABLE OF CONTENTS

page

LIST OF TABLES

LIST OF FIGURES

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

THREE ESSAYS ON PRODUCTION PLANNING IN SEMICONDUCTOR MANUFACTURING

By

Bülent Çatay

August 1999

Chairman: Dr. S. Selcuk Erenguc
Cochair: Dr. Asoo J. Vakharia
Major Department: Decision and Information Sciences

Semiconductor manufacturing is one of today's most complicated manufacturing processes. Most of the recent research in semiconductor manufacturing focuses on demand satisfaction, maximization of equipment utilization, and maximization of throughput with some capacity constraints. In this research, the first chapter presents an overview of the semiconductor manufacturing technology and describes the manufacturing process. It also discusses a real world production planning application in a major U.S. semiconductor company and proposes a new methodology for capacity planning.

The second chapter addresses the assignment problem of wafers to alternative machine groups and the determination of equipment requirements by recognizing

capacity limitations of the individual machines as well as reducing operating and investment costs related to the machines. A mathematical model, which is a variation of the well-known generalized assignment problem with continuous/integer variables and fixed charges, is developed for this problem. Given the intractability of the model, lower bounding and heuristic methods to solve the problem are proposed.

The third chapter extends the problem described in chapter 2 to a dynamic environment where demands vary between time periods and inventory carrying is allowed. We investigate the tradeoff between allocating a constant capacity for the planning horizon and holding inventory and we present a mixed-integer mathematical model to minimize operating and investment costs related to the machines and inventory holding costs. We propose a Lagrangian relaxation based heuristic approach to solve the problem.

The fourth chapter presents an effective algorithm to identify a set of component families in order to effectively to sequence individual printed circuit boards to multiple, identical insertion machines. The objective of the component grouping problem is to maximize the sum of similarities between each component type and a designated component type chosen as the group median in an attempt to minimize the number of machines a printed circuit board needs to visit. Once the component families are formed, two objectives are considered in scheduling the printed circuit boards: minimizing the maximum completion time (makespan) and the mean flow time. Finally, in the fifth chapter we discuss the implications and conclusions of this study and give directions for future research.

CHAPTER 1
INTRODUCTION

1.1. Semiconductor Manufacturing

This chapter presents an overview of the semiconductor manufacturing technology, describes the manufacturing process introducing the terminology that will be used in the following chapters. It also discusses a real world production planning application in a major U.S. semiconductor company and proposes a new methodology for capacity planning.

1.1.1. Semiconductor Industry Overview

The semiconductor industry is an excellent example of growing and innovative industry. Almost non-existent 40 years ago, this industry has grown steadily since 1985 and particularly fast since 1992, with annual growth exceeding 30 percent. Industry wide revenues of $27 billion in 1984 and $142 billion in 1996 were reported. Projections are that these will be $425 billion by 2001. This rapid growth of the semiconductor industry has been fueled by many factors, including:

- A PC market that continues to grow – In 1995, the demand for PCs resulted in an estimated 38 percent of chip market consumption.

- The use of semiconductors in a wider variety of products – Semiconductors are now used in products such as electronic equipment, appliances, cellular phones, and automobiles. Consumers use chips embedded in goods ranging from the microwave oven to the garage-door opener.
- The increasing microelectronics content of electronics products – Chip content of the average electrical product was 4.3 percent in the 1970s, 6.8 percent in the 1980s, and 10.2 percent in the early 1990s.
- Emerging international markets, including Korea, Taiwan, and China – Today, these regions represent 20 percent of world consumption, compared to 7 percent a decade ago.

Such a rapid growth in the semiconductor industry requires a continuous expansion of the manufacturing infrastructure. Much of the current restructuring of the U.S. semiconductor industry may be understood in terms of efforts to reduce cost while producing more powerful chips. The success in these efforts has been largely based upon innovation and technology developments such as the implementation of more advanced photolithography techniques and the utilization of larger diameter silicon wafers.

However, the productivity gains based only on product and process developments will not be sufficient to offset the vigorous growth in manufacturing costs. In terms of semiconductor-fabrication equipment, the catch up between new manufacturing techniques and the development of corresponding equipment has clearly become a never-ending race in which each surpasses the other. Manufacturers have been confronting price

increases of 28 percent per year in machinery. This, in turn, is raising the high cost of building new state-of-the-art semiconductor facilities and investing on new equipment.

The Semiconductor Industry Association reports that new fabrication plants (called "fabs") cost an average of $2.5 billion. Within the next two years, the biggest fabs will cost more than $5 billion. Once built, fabs can become obsolete in 3 years to 5 years. In the meantime, market demand for chips can change rapidly, making it difficult for manufacturers to adjust production schedules to market demands. This problem is being partially alleviated by the fact that chipmakers are now tapped into networks that link them to their PC customers' inventory systems. As a result, they can use just-in-time manufacturing techniques to match supply with demand.

However, with line widths (the widths of individual electronic components etched onto the wafer) now being cut to fractions of a micron, chipmakers still may face diminishing returns on their next generation of fab manufacturing investments. Although the cost of a fabrication plant doubled between 1984 and 1990, chipmakers were still able to double the number of transistors on a chip. With today's shrinking line widths, however, it will be difficult for the next generation of fabs to double chip performance again. While focusing on the development and deployment of new technologies, semiconductor companies must simultaneously embrace other methods to increase their productivity and to remain globally competitive. These methods must involve improvement of capital utilization through the increase of overall equipment effectiveness and reduction of cycle times and work-in-process inventories.

1.1.2. Semiconductor Technology Overview

Semiconductors, or silicon components, are the basic building blocks of every piece of electronic equipment. Essential in computers, they are also part of everything from microwave ovens to automobiles. The components themselves are small, flat chips of silicon with miniaturized electronic circuits etched onto them. The element silicon is the most commonly used semiconductor. When impurities are added to it (a process called doping), silicon can become a conductor or insulator, depending on the dopant used. The term semiconductor is also used in a broader sense today to refer to electronic components fabricated from semiconductor materials, such as integrated circuits, processors, and auxiliary chips.

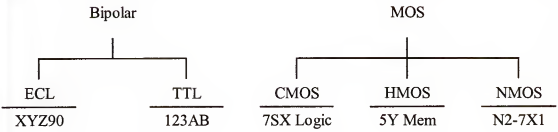
The most common natural semiconductors used in electronics are the elements of silicon and germanium. Silicon, one of the elements of common beach sand, is the most frequently used semiconductor material because it can be used at a wide variety of temperatures. Silicon is also the second most abundant element on earth, after oxygen. Semiconductors can be classified according to their material (silicon, germanium, etc.), their product groups (discrete devices, integrated circuits, etc.), their technology (bipolar, MOS), their capability to handle power, and their customization (standard, custom and semicustom). We will be focusing on the classification of integrated circuits by their technology.

In the semiconductor industry, the processes to make integrated circuit (IC) products may vary from one company to another, yet there are usually some strong

similarities. By design technology, there are only a few major groups or divisions such as MOS (Metal Oxide Semiconductor) and Bipolar. MOS transistors are small and consume relatively little power, but historically were relatively slow. Circuits built from bipolar transistors are larger and faster, but consume a great deal more power. Over the past decade, MOS technology became more sophisticated, resulting in increased processor speed.

These two divisions have usually subdivisions. CMOS (Complementary MOS) is now the most frequently used technology in systems ranging from PCs to mainframes to supercomputers. Each of these subdivisions has its own method for making its products. Within each subdivision, many products will be manufactured. An ECL (Emitter Coupled Logic) group will not produce just one type of product, in fact, they will possibly make hundreds of different products.

In semiconductor manufacturing, a combination of numbers and letters is used to identify a particular product type that is being made. Similar to the automobile industry where companies (e.g. Chrysler) have different divisions (e.g. Dodge, Plymouth, Jeep) that manufacture cars with different names (e.g. Stratus, Neon, Cherokee), computer chips in semiconductor industry have number/letter codes as illustrated in Figure 1.1. The major difference between the automobile industry and semiconductor industry is that General Motors does not manufacture Neons nor does Chrysler manufacture Cavaliers. On the other hand, in semiconductor industry many companies may be making the same XYZ90 product.



ECL: Emitter Coupled Logic
TTL: Transistor-Transistor Logic
CMOS: Complimentary Metal Oxide Semiconductor
HMOS: High density Metal Oxide Semiconductor
NMOS: N-channel Metal Oxide Semiconductor

Figure 1.1. Classification of integrated circuits[1]

1.1.3. Semiconductor Manufacturing Process

Manufacturing a computer chip is a complex process involving hundreds of steps and requiring from a few days to up to three months of processing time to complete. Chips require incredible precision, careful attention to detail, and constant testing. The manufacturing process can be divided into five stages: the mask design, the crystal growth and slice preparation, the fabrication of the device, the assembly and the final testing.

[1] All integrated circuit names are fictitious.

OXIDATION	DIFFUSION	DEPOSITION	MASKING
Initial Oxidation	N+ Diffusion	1st Deposition	1st Mask
Boron Oxidation	Isolation Diffusion	2nd Deposition	2nd Mask
3rd Oxidation	P+ Diffusion	3rd Deposition	Metal Mask

Figure 1.2. Semiconductor manufacturing operation steps

In IC manufacturing, each letter and number code represents a distinct product and each product has its own special set of operations such as "Oxidation," "Diffusion," "Masking," etc. which must be performed to make it successfully function. Some of the work performed in these operations are shown in Figure 1.2. These processing steps are combined into a certain work flow order when the wafers are processed and each particular sequence throughout its fabrication process visiting different work centers. Specifically, each wafer makes multiple visits to the same work center at different points in the fabrication process. The common practice in the layout of facilities for different types of processes is to group similar operations together: furnaces are grouped in one area, ion implanters in another, and so on. This layout requires wafers moving back and forth between work centers but it allows the utilization of the same equipment to process wafers at different operation steps. For instance, all masking operations may be completed on the same machine or in the same work center although the wafers are required to travel to other processing areas between these masking operations.

Table 1.1 illustrates an example of operation naming and sequencing practice along with the production volume of each product type[2]. Note that the operation steps may be same or distinct for each product type and the operation name indicates both the product type and the process to be performed on that product type. The product and operation incidences with the scheduled volume of products at each operation step are also given in Table 1.2 as an example. This example depicts the case of three types of wafer and five types of process they require during the manufacturing process: oxidation, masking, deposition, diffusion, and ion implantation. These processes are repeated at different steps and there are 21 distinct operation steps. The operation name describes the type of the process at each step and the wafer type to be processed. For example, {CM-OX1-X} can be interpreted as the first oxidation step on a X type CMOS wafer. The number of CMOS/X wafers to be processed each day at this stage is 80. Similarly, {CM-DEP1-XYZ} denotes the first deposition process for all X, Y, and Z types of CMOS wafers. The scheduled volume at this operation step is 300 wafers, which is the sum of the volumes of CMOS/X, CMOS/Y, and CMOS/Z chips, since they all require the same deposition process and can be processed with the same setup. Assuming 100 % yield per operation in this example, the number of wafers per day remain constant through the manufacturing process.

---

[2] The names are for illustrative purposes and fictitious.

Table 1.1. Operation names and sequences by products

PRODUCT NAME	VOLUME (Wafers/Day)	OPERATION NUMBER	OPERATION NAME
CMOS / X	80	1	CM-OX1-X
		4	CM-M1-XY
		6	CM-DEP1-XYZ
		7	CM-DIFF1-X
		10	CM-OX2-X
		12	CM-M2-XZ
		14	CM-IMP1-XZ
		16	CM-DEP2-XY
		18	CM-DIFF2-XY
		20	CM-IMP2-XZ
CMOS / Y	100	2	CM-OX1-Y
		4	CM-M1-XY
		6	CM-DEP1-XYZ
		8	CM-DIFF1-Y
		11	CM-OX2-YZ
		13	CM-M2-Y
		15	CM-IMP1-Y
		16	CM-DEP2-XY
		18	CM-DIFF2-XY
		21	CM-OX3-Y
CMOS / Z	120	3	CM-OX1-Z
		5	CM-M1-Z
		6	CM-DEP1-XYZ
		9	CM-DIFF1-Z
		11	CM-OX2-YZ
		12	CM-M2-XZ
		14	CM-IMP1-XZ
		17	CM-DEP2-Z
		19	CM-MM-Z
		20	CM-IMP2-XZ

Table 1.2. Product / Operation incidences and scheduled volumes

P R O D U C T S

OPERATIONS	CMOS / X	CMOS / Y	CMOS / Z
1st Oxidation	1	2	3
2nd Oxidation	10	11	11
3rd Oxidation		21	
….			
….			
1st Masking	4	4	5
2nd Masking	12	13	12
3rd Masking			
Metal Masking			19
….			
….			
1st Deposition	6	6	6
2nd Depostion	16	16	17
….			
….			
1st Diffusion	7	8	9
2nd Diffusion	18	18	
….			
….			
1st Ion Implant	14	15	14
2nd Ion Implant	20		20
….			
….			

Operation No	Operation Name	Volume (wafers/day)
1	CM-OX1-X	80
2	CM-OX1-Y	100
3	CM-OX1-Z	120
4	CM-M1-XY	180
5	CM-M1-Z	120
6	CM-DEP1-XYZ	300
7	CM-DIFF1-X	80
8	CM-DIFF1-Y	100
9	CM-DIFF1-Z	120
10	CM-OX2-X	80
11	CM-OX2-YZ	220
12	CM-M2-XZ	200
13	CM-M2-Y	100
14	CM-IMP1-XZ	200
15	CM-IMP1-Y	100
16	CM-DEP2-XY	180
17	CM-DEP2-Z	120
18	CM-DIFF2-XY	180
19	CM-MM-Z	120
20	CM-IMP2-XZ	200
21	CM-OX3-Y	100

1.2. A Production Planning Application

This section presents a production planning application in a major U.S. semiconductor company that develops, manufactures, and markets semiconductor technologies, products, packaging and services for their internal and external customers. The manufacturing facilities range from low volume, high product variation research and development lines to high volume lines with both low and high volume variation.

The manufacturing facility where this study has been conducted has two wafer fabrication areas. These fabs have been connected to each other recently to facilitate the routing of work-in-process (WIP) inventories and provide possible use of additional machine capacity. The total number of machines in these fabs is around 1200. The cost of a machine varies roughly between $2 million and $7 million and the facility purchases 50 machines on the average each year to replace the obsolete machines or to add capacity to meet demand forecasts. Hence, capacity planning is a strategic decision since the cost of investing in new equipment is significantly high.

1.2.1. Machine Planning System Overview

The Machine Planning System (MAPS) is designed to facilitate manufacturing capacity planning, which relates to estimating the amount and type of process equipment required to produce a given product mix over a particular time, and using these estimates for current managerial decisions and new facility construction. When only a unique machine can complete a certain operation this calculation is relatively straight forward;

simply sum the number of operations to be processed, divide by the production capacity of the machine type, its yield, and its utilization to arrive at the number of machines required. However, in most manufacturing facilities there are often several different machine sets that can be used to process a particular operation. In the case where no single machine group has the sufficient capacity to process all the required volume, the production WIP needs to be allocated among alternate machine groups and the capacity across all possible machine groups is assessed.

Each facility in this company uses its own internally developed Machine Planning System and each system is linked to others' databases (However, the fabrication areas remain independent). The major drawbacks of the present system are the incompatibility

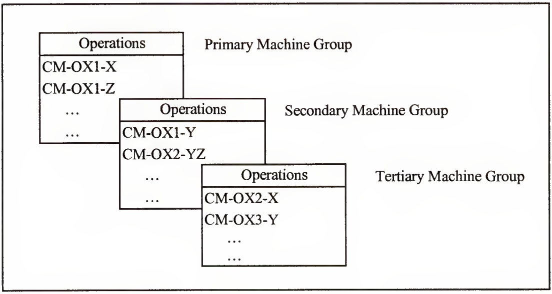


Figure 1.3. Cascades in MAPS

of one system with another, which causes difficulties in data integration and exchange, and the age which results in extensive and expensive maintenance and upgrade requirements. The machine planning engine in MAPS deals with groupings of similar machines called "machine groups". Special names are assigned to different machine groups and all machines within a group are equivalent. Machine groups are allowed to be dedicated to different operations but not individual machines within a group.

1.2.2. Cascading in MAPS

Cascading is the term used for allocating production work across alternate parallel machine groups in capacity planning. Given a manufacturing line, there is typically a mix of multiple machine types with different production characteristics that can perform the same operation. One machine type is generally preferred over another, though this preference may differ according to either the process or product involved. As a result, priorities are assigned to each unique process/product which we define as an operation and work is allocated across alternate machine groups on that basis.

The machine groups in MAPS are classified as primary machine group, secondary machine group, tertiary machine group, and so on, for any operation set where primary machines are the most efficient machines that can process the specified operations and secondary, tertiary, etc. machines are the alternative machines in case additional machine capacity is needed. Cascading in MAPS refers to the allocation of secondary machine capacity to operations whose primary machine capacity has been exceeded.

In the current system, WIP allocation is carried out sequentially. In general terms, the machine groups that can process some of the same operations (product/process combinations) are assigned in ordered sequence to a cascade group. The production volumes for all operations assigned to the cascade group are first allocated to the primary machine group, once its capacity is reached the remaining volumes are allocated to the secondary machine group, then to the tertiary, etc., until all machine groups have been allocated with as much work as their defined capacity. If unsolutioned volumes remain after loading all machine groups in the cascade groups, the remaining work is allocated first to the primary machine group, for all operations defined on that machine group, if not possible, then sequentially to the secondary and tertiary machine groups, etc., until all production volumes have been assigned. Thus, if the primary machine group is defined with the capability of processing all operations within the cascade group, all unsolutioned volume would be allocated to just that single machine group, leaving it the only machine group needing additional machines to contain the capacity requirement. The term cascading comes from the analogy of describing each machine group as a pool of capacity, the production volumes flow first into the primary pool, once it is full the remaining volume (if any) flows to secondary pool, and so on, like a cascading waterfall. A step by step description of the algorithm is as follows:

Current Cascading Algorithm:

Step 0 : Compute the number of machines required for each operation.

Step 1 : Assign all unique operations to machine groups that are capable of performing them.

Step 2 : Fill up primary machine group in the given order of the operations.

Step 3 : Repeat with secondary machines if primary machines' capacity has been exceeded.

Step 4 : Repeat until all cascaded machines are used up or all operations are assigned.

Step 5 : Add machines to satisfy demand (if needed).

Step 6 : Repeat for all periods..

The data required for the algorithm include daily production capacity, yield, and utilization of each machine type, number of machines of each type in each machine group and scheduled volume per day of each product for the specified operation steps. The production capacity is the number of wafers processed per day and is implicitly a function of the raw processing time, setup time, and operator intervention time. The yield is the ratio of number of good wafers at the end of the process to the number of wafers that go in. Utilization is the net production time of the machine group during the 24-hour period. Using this data, the number of machines required by each operation on each machine group is computed as follows based on the assumption that all wafers are processed on that machine group:

$$\text{Number of machines needed} = \frac{\text{Volume}}{\text{Production Capacity} * \text{Yield} * \text{Utilization}}$$

1.2.3. A New Methodology for Cascading

In this section we introduce a new cascading methodology that includes enhancements designed to improve some inherent limitations within the current MAPS cascading methodology. Primary among these is the restriction that all operations defined within the cascade group have the same ordinal allocation priority across the machine groups in a cascade set. Although one machine group is often preferred over another for a particular operation, this machine preference may vary among the designated operations due to some factors such as speed, yield, logistics, robustness, etc. Thus, a sequential WIP flow such as current cascading can result in an impractical and unrealistic capacity consumption solution. The new cascading algorithm introduces a declared allocation priority for each operation across the machine groups within the cascade set. It is a more flexible bi-directional WIP allocation methodology that should result in a more practical machine consumption solution.

At the time that an operation is associated with a machine group, whether cascaded or not, it is assigned a load priority sequence number. This priority number is a non-unique number describing a weighted preference for the order in which volumes should be processed through the machine. A priority value of 1 is the highest priority or first operation to be processed, a 9 would be the lowest priority or last operation processed on that machine group[3]. All operations with the same priority number are loaded in the original sequence.

---

[3] A single digit is used for illustrative purposes.

Once the cascade group is defined with sequentially ordered machine groups, an allocation priority number is assigned defining an operation's preference for each capable machine group. This priority is a non-unique number describing a weighted preference for where operation should be processed. Factors to consider include speed, yield, logistics, robustness, etc. A priority value of 1 designates the most preferred machine group, a 9 the least preferred for each operation.[4] If multiple machine groups are designated the same preference number, operations are assigned based upon the original sequence of machine groups within the cascade set.

After both priorities are defined, a composite priority is created for every operation on each machine group within the cascade set called as "cascade priority." This is a two-digit integer value where the right digit is the first priority number described above and the left digit is the other. If the operation cannot be processed by a machine group, the restricted priority of 99 is assigned.

Based on these cascade priorities the proposed methodology assigns operations across machine groups in two stages. In the first stage, the operations are sorted lowest to highest in order of their cascade priority number for each machine group. At the first pass, operations whose priorities start with 1 are assigned in the order of the second digit of their priorities starting with the first machine group in the order (primary machine group in the current algorithm) and followed by the second machine group (secondary machine group in the current algorithm), then third machine group (tertiary machine

[4] Again, a single digit priority is used for illustrative purposes.

group), and so on. Volumes are assigned until the individual machine group is at capacity for the number of available machines. Left over volumes are assigned either in the following passes on the other machine groups or at the end of the first stage as unsolutioned operations.

After all machine groups have been assigned with operations whose priorities start with 1, a second pass is performed in the same manner where operations whose priorities start with 2 are assigned in the order of the second digit of their priorities. This allocation procedure is continued until all volumes are assigned or all machine groups in the cascade set are at capacity for the number of available machines. Any remaining unsolutioned operation volumes are then assigned to their most preferred machine group (i.e. to machine group with the lowest cascade priority number).

By allocating the volumes based upon the cascade priorities it is possible that one or more machine groups in the cascade set be loaded over capacity while other machine groups are underloaded. Since this situation contradicts the primary goal of minimizing additional machine investment a second stage in the allocation is undertaken for operations on overloaded machine groups only. Operation volumes are reassigned from overloaded machine groups, based on reverse cascade priority, to underloaded machine groups.

In what follows is a step by step description of the proposed cascading methodology. A brief example is presented in the next section.

New Cascading Algorithm:

Step 0 : Compute the number of machines required for each operation.

Step 1 : Assign the operations to the first machine group in the non-decreasing order of cascade priorities that start with 1, until all machines are used up or all operation volumes are assigned.

Step 2 : Repeat with the second machine group and continue until the last machine group is considered.

Step 3 : Return to the first machine group for the second pass (considering cascade priorities between 21-29 this time) and repeat the same procedure with unassigned, partially or completely, operations. Stop after the procedure is executed for a predetermined number of passes.[5]

Step 4 : Compute machine shortages (if any) for unsolutioned operations and assign them to their most preferred machine groups.

Step 5 : If one or more machine groups are overloaded while there exist excess capacity on another (other) machine group(s), transfer operation(s) in the non-increasing order of the cascade priorities scanning all underloaded machine groups in reverse cascade priority.

Step 6 : Continue transferring operations until overloaded machine group(s) is (are) reduced to full capacity or underloaded machine group(s) is (are) loaded up to full capacity.

Step 7 : Repeat for all periods.

---

[5] The number of passes is set equal to the number of machine groups.

Step 8 : If additional machines are needed, add them and redo assignments following Step 1 through Step 3.

1.2.4. Cascading example

This example describes the production planning of 3 machine groups and 6 operations for 2 periods using the proposed cascading algorithm. There are 5 machines in the machine group 1 and 3 in machine groups 2 and 3. The scheduled processing volumes of wafers per day are 400 for all operations during the first period and 520, 500, 400, 400, 500, 520, respectively, during the second period. The yields of machine groups are 1.00, 0.99, 0.98, their production capacities are 300, 325, 333, and their utilizations are, 0.90, 0.80, 0.75, respectively. The cascade priorities are as shown on Table 1.3. The first digit is the preference of the machine group across all machine groups for the operation and the second digit is the assignment preference of the operation among all operations. For

Table 1.3. Priority table for the cascading example

	Oper 1	Oper 2	Oper 3	Oper 4	Oper 5	Oper 6
Machine Group 1	11	12	23	99	34	35
Machine Group 2	23	24	99	11	99	32
Machine Group 3	99	23	34	35	11	12

Table 1.4. Assignment of operations at each pass

	Machine Group 1	Machine Group 2	Machine Group 3
1st Pass →	Operation 1 Operation 2	Operation 4	Operation 5 Operation 6
2nd Pass →	Operation 3	Operation 1 Operation 2	Operation 2
3rd Pass →	Operation 5 Operation 6	Operation 6	Operation 3 Operation 4

instance, priority of operation 1 for machine group 1 is 11, which means that at the first pass this operation will be considered first to be assigned to machine group 1, then operation 2, which has priority 12, will be considered for the same machine group. At the same first pass, operation 4 will be assigned to machine group 2 (it has cascade priority 11) and operation 5 to machine group 3 (it has cascade priority 11). At the second pass, operation 3 will be assigned to machine group 1 if that machine still has capacity, operations 1 and 2 will be assigned to machine group 2 if they have not been completely assigned to machine group 1 and any remaining WIP of operation 2 will be assigned to machine group 3. The same procedure will be applied for a third pass. Note that we allow partial assignment of an operation to more than one machine group depending on the machine capacities. The assignment order in this procedure at each pass is given on Table 1.4.

Table 1.5. Production plan after the first stage

	Machine Group 1		Machine Group 2		Machine Group 3	
	Period 1	Period 2	Period 1	Period 2	Period 1	Period 2
Operation 1	1.48	1.93	0.00	0.00	0.00	0.00
Operation 2	1.48	1.85	0.00	0.00	0.00	0.00
Operation 3	1.48	1.48	0.00	0.00	0.00	0.00
Operation 4	0.00	0.00	1.55	1.55	0.00	0.00
Operation 5	0.00	0.00	0.00	0.00	1.63	2.04
Operation 6	0.24	0.00	0.00	1.11	1.37	0.96
Available	5	5	3	3	3	3
Required	4.68	5.26	1.55	2.66	3.00	3.00
Whole	5	6	2	3	3	3
Delta	0	-1	1	0	0	0
SHORTAGES						
Operation 3	0.00	0.26	0.00	0.00	0.00	0.00

Table 1.5 illustrates the production plan and additional machine requirements at the end of the first stage after three passes are completed. Notice that following its cascade priority unsolutioned WIP of operation 3 is assigned to machine group 1, resulting this machine group to be overloaded in the second period. Notice also that although machine group 2 is under capacity it cannot process operation 3 since the cascade priority of operation 3 for machine group 2 is 99.

Table 1.6. Final production plan

	Machine Group 1		Machine Group 2		Machine Group 3	
	Period 1	Period 2	Period 1	Period 2	Period 1	Period 2
Operation 1	1.48	1.93	0.00	0.00	0.00	0.00
Operation 2	1.48	1.59	0.00	0.28	0.00	0.00
Operation 3	1.48	1.48	0.00	0.00	0.00	0.00
Operation 4	0.00	0.00	1.55	1.55	0.00	0.00
Operation 5	0.00	0.00	0.00	0.00	1.63	2.04
Operation 6	0.24	0.00	0.00	1.11	1.37	0.96
Available	5	5	3	3	3	3
Required	4.68	5.00	1.55	2.94	3.00	3.00
Whole	5	5	2	3	3	3
Delta	0	0	1	0	0	0
SHORTAGES						

The final production plan following the second stage is shown on Table 1.6. Unassigned WIP of operation 3 remains assigned to machine group 1, its most preferred machine group, while a portion of operation 2 is transferred from machine group 1 to machine group 2 since the latter has unused capacity. Notice that transferred amount of “0.26” in machine group 1 corresponds to “0.28” in machine group 2 due to production capacity difference between the two machine groups.

1.3. The Organization of this Dissertation

This rest of this dissertation consists of three chapters with each chapter being an independent essay and complete within itself. The first two chapters address the problem of operation/machine type assignment and machine type duplication in semiconductor manufacturing as an introduction is presented in this chapter. Chapter 2 introduces a mixed integer programming model of the problem considering the operating and procurement costs of the machines and provides efficient lower bounding and heuristic methods to solve the problem. Chapter 3 is an extension of this problem to a multi-period production planning perspective where inventories may be carried over periods. Chapter 4 proposes an effective algorithm to identify the component families from a large number of component types in the manufacture of printed circuit board assembly system and to schedule the boards to insertion machines. Finally, Chapter 5 summarizes this study and gives directions for future research.

CHAPTER 2
OPERATION/MACHINE TYPE ASSIGNMENT IN SEMICONDUCTOR MANUFACTURING

2.1. Introduction

Manufacturing a computer chip is a complex process involving hundreds of steps and requiring from a few days to up to three months of processing time to complete. The manufacturing process consists of various operation types such as oxidation, deposition, lithography, diffusion, etching, ion implantation, etc. Some of these operation types are repeated several times to build different layers on the wafer. Each wafer type follows a particular sequence throughout its fabrication process visiting different work centers. Specifically, each wafer makes multiple visits to the same work center at different points in the fabrication process. The common practice in the layout of work centers for different types of processes is to group similar operation types together: furnaces are grouped in one area, ion implanters in another, and so forth. This layout requires wafers moving back and forth between work centers but it allows the utilization of the same equipment to process wafers at different steps. For instance, all oxidation processes may be completed on the same machine or in the same work center although the wafers are required to travel to other work centers between these oxidation process steps.

The capacity allocation and machine duplication in semiconductor manufacturing involves the assignment of a set of wafers requiring certain process types to different machine groups, and the determination of equipment requirements. We represent the problem of assigning operations to different machine groups where machine duplication is permitted as a variation of the well known Generalized Assignment Problem (GAP). Given a set of tasks to be assigned to a group of agents and the cost of performing each task by each agent, the GAP consists of minimizing the cost of assigning tasks to agents such that each task is assigned exactly to one agent subject to the availability of a single resource type that is consumed by the agents when performing these tasks (Ross and Soland, 1975).

The applications of GAP include computer job assignments in computer networks, assignment of software development tasks to programmers, scheduling variable length commercials into time slots, scheduling payment on accounts where lump sum payments are prespecified (Balachandran, 1972), assignment of ships to yards for overhaul (Gross and Pinkus, 1972), communication network design models with certain node capacity constraints (Grigoriadis, Tang, and Woo, 1974). Ross and Soland (1977) show how to formulate and solve some special facility location problems as GAPs. Fisher and Jaikumar (1981) address the vehicle routing problem based on a GAP model where the vehicle fleet delivers products from a central warehouse to scattered customers. An extension to GAP is known as the Multi-Resource Generalized Assignment Problem (MRGAP) where the agents consume multiple resources when performing their tasks (Gavish and Pirkul, 1991). MRGAP was used in the design of distributed computer

systems (Gavish and Pirkul, 1986) and in trucking industry (Murphy, 1986). The interested reader is referred to Cattrysse and Van Wassenhove (1992) for a broad survey of the algorithms developed for GAP and MRGAP.

Both GAP and MRGAP assume that each task is assigned to a single agent. However, in semiconductor industry, as well as in many other industries, it is a common production planning practice to assign a task to two or more agents. In this chapter, we discuss an operation/machine type assignment problem where task splitting is allowed and resource expansion is possible. The remainder of the chapter is organized as follows. In Section 2.2 we describe the problem and formulate a mathematical model. In Section 2.3 we discuss Lagrangian relaxation and Lagrangian decomposition techniques to obtain lower bounds to the problem, and a solution procedure for the problem is described in Section 2.4. In Section 2.5, we describe experimental analysis of the proposed solution method on both randomly generated and industry data.

2.2. Problem Description and Formulation

We model the problem of assigning individual operations to predetermined machine groups where machine duplication is allowed as a variation of GAP with both continuous and integer variables and fixed charges. Unlike the binary decision variables in GAP, which restricts the assignment of an operation to only one machine group, partial assignments of operations to several machine groups is permitted in our problem. The machine capacity constraints include integer decision variables to determine additional machine requirements in order to meet the production schedule. Since the number of

additional machines is not restricted, a feasible solution to the problem always exists. The fixed costs imposed on the procurement decision variables relate the problem to a Fixed Charge Transportation Problem as well. Given the number of machines in each machine group and capacity consumption of each operation type in each machine group, our objective is to find the assignment scheme of individual operations to machine groups that minimizes the total monthly operating cost of machines processing these operations and the procurement cost of additional machines capitalized monthly. To the best of our knowledge no research exists in the literature that addresses this type of problem with integer and continuous variables involving both assignment and capacity decisions. Only, Askin, Selim, and Vakharia (1997) approach a similar problem in the operations assignments phase of their methodology for designing flexible cellular systems where the resource capacities may vary. They present two greedy heuristics to find the resource capacity. Since they do not permit partial assignments, their problem reduces then to the GAP.

As we mentioned earlier, certain operation types are repeated several times during the manufacturing process, allowing wafers to visit the same machine group more than once. Since the specifications of the process performed on the wafer differ from one visit to another, each visit is referred to as a distinct step and each processing step is given a specific name. This name indicates both the wafer type/s and the process to be performed on the wafer type/s. Throughout this chapter wafer type/s at a particular processing step is/are referred to as an "operation". In this environment, we focus on the assignment problem of operations to alternative machine groups and we consider a population of

machines that are only capable of processing these operations. Machines have different manufacturing characteristics according to the process and product involved. While newer machines are normally capable of processing advanced products as well as older products, older machines can only process older products, may require longer processing times and may have lower yields and lower utilization levels due to more maintenance requirements. Machines having the same characteristics are gathered into machine groups and each machine group represents a set of parallel identical machines with the same processing capabilities. A machine group may be constituted of one or more machines in an existing manufacturing system. The number of machines is zero in the case where a new machine group is formed. In what follows a discussion of the data requirements and mathematical formulation of the assignment problem is presented.

2.2.1. Data Requirements

The set of machine groups, $I = \{1, \ldots, m\}$, set of operations, $J = \{1, \ldots, n\}$, and the number of machines in each machine group, N_i, are given. For an existing manufacturing system, N_i is the number of machines currently available in each machine group i. For a new machine group i, N_i is zero.

A matrix $[P]$ is defined as the capability matrix: the binary entry p_{ij} is 1 if machine group i can perform operation j and 0 otherwise. The operations may be split among machine groups and can be performed in any order. Daily scheduled volume of operations (V_j), the yield of machine type i with respect to operation j (Y_{ij}), and the

Table 2.1. Illustrative data for four operations processed on ion implanters

Machine Type	Quantity Available	Net Production Hrs/24 Hr	Utilization
Ion Implanter 163	3	19.2	0.80

Operation Number	Scheduled Volume	Production Capacity	% Yield	Number of machines needed
5	95	544	98.12	0.18
11	244	1022	95.26	0.25
16	169	898	92.96	0.20
23	44	643	96.37	0.07

quantity of operation j (in other words, number of wafers) that machine group i can process per day (PC_{ij}) are assumed to be known. PC_{ij} is computed as a function of the raw processing time, setup time, and operator intervention time. Utilization (U_i) is the net production hours of the machine group i during the 24 hour period. Given these data, the number of machines of type i required to process operation j (α_{ij}) is computed with the following formula, assuming that all machines are available 100% of the time:

$$\alpha_{ij} = \frac{V_j}{(PC_{ij})\,(Y_{ij})} \cdot p_{ij}$$

The data for four operations are displayed in Table 2.1 as an example. Also shown is a machine group consisting of three ion implanters that can process these operations. The

monthly operating cost of each machine of type i to process operation j (c_{ij}) and monthly capitalized procurement cost of an additional machine of type i (F_i) are also known.

2.2.2. Problem Formulation

A comprehensive mathematical model of the operation-machine type assignment problem with integer and continuous decision variables and fixed charges on resources is as follows:

$$\text{(MIAP)} \quad \min \quad \sum_{i \in I} \sum_{j \in J} c_{ij} \alpha_{ij} x_{ij} + \sum_{i \in I} F_i Q_i \tag{1}$$

$$\text{s.t.} \quad \sum_{i \in I} x_{ij} = 1, \qquad j \in J \tag{2}$$

$$\sum_{j \in J} \alpha_{ij} x_{ij} \leq (N_i + Q_i) U_i, \qquad i \in I \tag{3a}$$

$$0 \leq x_{ij} \leq p_{ij}, \qquad i \in I, j \in J \tag{4}$$

$$Q_i \geq 0 \text{ and integer}, \qquad i \in I \tag{5}$$

The decision variable x_{ij} ($0 \leq x_{ij} \leq 1$) indicates the fraction of operation j assigned to machine group i and integer variable Q_i represents the number of additional machines required in machine group i. α_{ij} is the fraction of machine type i required to perform operation j, assuming that all of operation j is processed on machine group i. U_i can be used as a parameter to control the amount of slack capacity built into the system ($0 < U_i \leq 1$). For a given U_i we define $a_{ij} = \alpha_{ij} / U_i$ and replace constraint (3a) with

$$\sum_{j \in J} a_{ij} x_{ij} - Q_i \leq N_i\,, \qquad i \in I \qquad (3b)$$

In the formulation of the Mixed Integer Assignment Problem (MIAP), the objective function (1) minimizes total monthly operating cost of machine groups and monthly procurement cost of additional machines. The first constraint set (2) enforces all scheduled volume for an operation to be completely assigned. Constraint set (3b) ensures that there is sufficient machine capacity in each machine group to process the assigned operations considering machine utilization and machine duplication. Constraints (4) prevent assignments of operations to machine groups that are not capable of processing them and constraints (5) impose integrality restriction on the number of additional machines. Hence, this model finds the optimum assignment scheme of operations to machine groups to minimize total operating and procurement costs of machines.

In the next section, we apply Lagrangian relaxation and Lagrangian decomposition techniques with subgradient optimization to MIAP to obtain lower bounds on the optimal solution. Lagrangian relaxation offers an alternative to linear programming relaxation for finding good bounds on the optimal solutions. Lagrangian decomposition is an extension to Lagrangian relaxation approach and may yield improved bounds on the optimal solutions. An efficient solution procedure is also developed to find a good assignment scheme for the problem.

2.3. Lower Bounding Methods

Lagrangian relaxation and Lagrangian decomposition techniques are discussed in this section to obtain good lower bounds on the optimal solution to MIAP. These two relaxation approaches have been successfully employed by numerous researchers in the past in many integer/mixed integer programming applications.

2.3.1. Lagrangian Relaxation

Lagrangian relaxation is a well known way to circumvent the direct consideration of "complicating" constraints in an optimization model and has been widely used in large scale mathematical programming. Lagrangian relaxation is formed by multiplying the complicating constraints with corresponding penalty costs (Lagrangian multipliers) and including them into the objective function. By dualizing those constraints, we obtain a problem that is easy to solve and whose optimal value gives a lower bound on the optimal value of the minimization problem. Although the method does not guarantee that a solution will be found for all problems, it is often worth trying since it is so simple compared to other methods. The reader is referred to Geoffrion (1974) and Fisher (1981) for a detailed and excellent study on the Lagrangian relaxation technique. Cattrysse and Van Wassenhove (1992) provide an extensive survey of several Lagrangian relaxation methods developed in the literature to solve the GAP.

The relaxation to our assignment problem is obtained by dualizing constraint set (2). The yielding Lagrangian relaxation LR(λ) with Lagrangian multiplier vector λ has the objective function

$$\text{minimize} \qquad \sum_{i\in I}\sum_{j\in J}(c_{ij}\alpha_{ij} - \lambda_j)\,x_{ij} + \sum_{i\in I} F_i Q_i + \sum_{j\in J} \lambda_j$$

where λ is unrestricted in sign. This problem nicely decomposes into m continuous knapsack subproblems, one for each machine group i. A procedure to solve this set of problems is as follows:

Step 1: Order x_{ij}'s in the increasing order of (c_{ij} α_{ij} - λ_j) / a_{ij} value.

Step 2: Proceed along the list: if (c_{ij} α_{ij} - λ_j) / a_{ij} is positive or zero, the corresponding x_{ij} is set equal to 0. If (c_{ij} α_{ij} - λ_j) / a_{ij} is negative, the corresponding x_{ij} is set to equal 1 assigning the whole production volume to machine group i, provided there is enough capacity left at that machine group, taking into account the other operations who have already been assigned to machine group i. However, if there is not sufficient capacity at machine group i to assign operation j completely, set x_{ij} as high as possible to exhaust capacity at machine group i.

Step 3: Compare the trade-off between the cost of adding an additional machine capacity and assigning remaining operations following the procedure in Step 2. If the difference is negative, increase Q_i by 1 and assign operations as described in Step 2.

Step 4: Repeat Step 3 incrementing Q_i until the objective function value ceases to decrease.

The preliminary analysis of this relaxation showed that the structure of the objective function coefficients largely affects the optimal value and weak lower bounds may be attained in some instances. Therefore, we considered adding a lower bound on the total machine requirements over all machine groups and dualizing it along with the capacity constraints. The new constraint is as follows:

$$\sum_{i \in I} Q_i \geq K, \tag{6}$$

We observed that this relaxation method could substantially improve the lower bound. The integer scalar K can be obtained using the following two methods:

- Solve the linear programming problem that minimizes total number of additional machines required subject to (2)-(4) and round up the objective function value.
- Consider the best case scenario: assign each operation j to the machine group i where it consumes least amount of resources regardless of the machine group's capacity but subject to its capability. Identify the number of additional machines required in each machine group i. Round up the difference between the total number of machines needed and total number of machines currently available.

Lagrangian relaxation LR(λ,μ), λ unrestricted and $\mu \geq 0$, with respect to constraints (2) and (6) is as follows:

$$\text{LR}(\lambda,\mu) \quad \min \quad \sum_{i\in I}\sum_{j\in J}(c_{ij}\alpha_{ij} - \lambda_i)x_{ij} + \sum_{i\in I}(F_i - \mu)Q_i + \sum_{j\in J}\lambda_j + K\mu \qquad (7)$$

$$\text{s.t.} \quad \sum_{i\in I} a_{ij}x_{ij} - Q_i \leq N_i, \qquad i \in I \qquad (8)$$

$$0 \leq x_{ij} \leq p_{ij}, \qquad i \in I, j \in J \qquad (9)$$

$$Q_i \geq 0 \text{ and integer}, \qquad i \in I \qquad (10)$$

We use a subgradient optimization procedure to update the Lagrangian multipliers. Given initial values of λ^0 and μ^0, we generate a sequence of Lagrangian multipliers λ^t and μ^t through the addition of a direction vector which is multiplied by a step size θ^t where θ^t is a positive scalar. Hence, the updating of Lagrangian multipliers of the constraint set (2) is done according to the equation

$$\lambda_j^t = \lambda_j^{t-1} + \theta^t\left(1 - \sum_{i\in I} x_{ij}\right), \qquad j \in J$$

and Lagrangian multiplier of the constraint (6) is updated according to the equation

$$\mu^t = \max\left\{0;\mu^{t-1} + \theta^t\left(K - \sum_{i\in I} Q_i\right)\right\}$$

The step size θ^t is updated at each iteration t using the following equation:

$$\theta^{t} = \delta^{t} \frac{(\mathrm{UB} - \mathrm{LB}(\lambda^{t-1}, \mu^{t-1}))}{\sqrt{\sum_{j \in J} \left(1 - \sum_{i \in I} x_{ij}\right)^{2} + \left(K - \sum_{i \in I} Q_{i}\right)^{2}}}$$

The step size θ^t depends on the parameter δ^t ($0 < \delta^t \leq 2$), on the gap between the current lower bound LB(λ^{t-1}, μ^{t-1}) and the estimated minimum value of the objective function of the relaxed problem, which is approximated by the upper bound UB obtained by applying a heuristic method, and on the Euclidean norm of the deviations in the relaxed constraints (2) and (6). The sequence δ^t is determined by setting $\delta^0 = 2$ and by halving δ^t whenever LB(λ^{t-1}, μ^{t-1}) does not increase after a fixed number of iterations. We terminate this procedure when one of the following stopping criteria is met:

(1) An iteration number limit,

(2) Maximum gap between the lower and upper bounds,

(3) A limit on the value of the Euclidean norm of the deviations.

2.3.2. Lagrangian Decomposition

The Lagrangian decomposition approach was introduced by Jörnsten and Näsberg (1986). The idea behind is that constraints sets (2) and (3b) are totally different, but both sets of constraints have a special structure. The traditional Lagrangian relaxation dualizes one of the constraint set and keeps the other. However, we may be able to obtain better lower bounds on the optimal objective function value by keeping both constraint sets in

the formulation. Consider the following reformulation of MIAP with the lower bounding restriction K on the sum of the total number of additional machines required:

$$
\begin{array}{llll}
\text{(AMIAP)} & \min & \beta \sum_{i\in I}\sum_{j\in J} c_{ij}\alpha_{ij}x_{ij} + \gamma \sum_{i\in I}\sum_{j\in J} c_{ij}\alpha_{ij}x_{ij} + \sum_{i\in I} F_i\, Q_i & (11) \\
 & \text{s.t.} & \sum_{i\in I} y_{ij} = 1, \qquad\qquad j \in J & (12) \\
 & & \sum_{i\in I} a_{ij}x_{ij} - Q_i \le N_i\,, \qquad i \in I & (13) \\
 & & \sum_{i\in I} Q_i \ge K\,, & (14) \\
 & & x_{ij} = y_{ij}\,, \qquad i \in I, j \in J & (15) \\
 & & 0 \le x_{ij}, y_{ij} \le p_{ij}\,, \qquad i \in I, j \in J & (16) \\
 & & Q_i \ge 0 \text{ and integer}, \qquad i \in I & (17)
\end{array}
$$

where β and γ are nonnegative parameters. MIAP and AMIAP are equivalent since the objective function values are related in the following way:

$$v(\text{MIAP}) = (\,\beta + \gamma\,)\; v(\text{AMIAP})$$

where v(P) is the objective function value of problem (P). Notice that v(MIAP) = v(AMIAP) when $\beta + \gamma = 1$. When the constraint sets (14) and (15) are relaxed, Lagrangian relaxation separates nicely into two problems, one problem with the x-variables only and another with y-variables only:

$$\text{LD}x(\lambda,\mu) \quad \min \quad \sum_{i\in I}\sum_{j\in J}(\beta\ c_{ij}\alpha_{ij} - \lambda_{ij})x_{ij} + \sum_{i\in I}(F_i - \mu)Q_i \tag{18}$$

$$\text{s.t.} \quad \sum_{i\in I} a_{ij}x_{ij} - Q_i \le N_i\,, \qquad i \in I \tag{19}$$

$$0 \le x_{ij} \le p_{ij}\,, \qquad i \in I, j \in J \tag{20}$$

$$Q_i \ge 0 \text{ and integer,} \qquad i \in I \tag{21}$$

$$\text{LD}y(\lambda) \quad \min \quad \sum_{i\in I}\sum_{j\in J}(\gamma\ c_{ij}\alpha_{ij} + \lambda_{ij})y_{ij} \tag{22}$$

$$\text{s.t.} \quad \sum_{i\in I} y_{ij} = 1\,, \qquad j \in J \tag{23}$$

$$0 \le y_{ij} \le p_{ij}\,, \qquad i \in I, j \in J \tag{24}$$

where μ is the nonnegative Lagrangian multiplier vector for constraint (14) and λ, unrestricted, is the Lagrangian multiplier vector for constraint set (15). Then, the objective function value of AMIAP is

$$v(\text{AMIAP}) = v(\text{LD}x(\lambda,\mu)) + v(\text{LD}y(\lambda)) + K\mu$$

LD$x(\lambda,\mu)$ decomposes into m continuous knapsack problems, LD$x_i(\lambda,\mu)$, which can be solved using the procedure described in the previous section. LD$y(\lambda)$, on the other hand, can be separated into n trivial generalized upper bound problems, LD$y_j(\lambda)$, where the assignment decision variables are naturally binary. The solution procedure is as follows: For all $j \in J$, determine $\text{argmin}_{i\in I}\ \{\gamma\, c_{ij}\ \alpha_{ij} + \lambda_{ij}\}$ and set the associated $y_{ij} = 1$.

Remaining y_{ij}'s are set to zero. Lagrangian multipliers in both problems are updated using the subgradient optimization procedure as described in Lagrangian relaxation method.

2.4. A Solution Procedure for Operation-Machine Type Assignment Problem

In this section, we propose a four phase solution procedure to the Operation-Machine Type Assignment Problem. An outline of the procedure is given in Figure 1. In Phase I we investigate linear programming relaxation of the problem. If the variables corresponding to machine duplication decisions are zero or integer then LP solution is optimal. Since this is very rarely the case we use the LP relaxation to obtain a lower bound on the objective function and an initial solution for the improvement procedure. Furthermore, if the problem is feasible with the number of additional machines set equal to zero, the solution gives an upper bound on the value of the objective function.

7

PHASE I:	Solve LP relaxation of MIAP (LP1). If the solution to LP1 is feasible to MIAP then it is also the optimal solution, stop. Otherwise, LP1 solution gives a lower bound to v(MIAP). If linear programming problem (LP2) where $Q_i = 0$ for $\forall i \in I$ is feasible then the optimal solution value is an upper bound to v(MIAP).
PHASE II:	Use a greedy heuristic to reduce the cost
PHASE III:	Use Lagrangian relaxation (LR) and/or Lagrangian decomposition (LD) method to improve the lower bound.
PHASE IV:	Solve linear programming problem for fixed values of Q_i obtained in Phase II to find the optimal assignment scheme and update the upper bound.

Figure 2.1. Operation-Machine Type Assignment Methodology

Phase II is a greedy heuristic consisting of an improvement procedure where we consider pair wise exchanges. Starting from an initial feasible solution, the procedure evaluates all possible exchanges between pairs of machines and makes an exchange if it improves the value of the objective function. We select the exchange providing the maximum cost improvement. The initial solution is obtained by solving the LP relaxation of the problem and rounding up the decision variables corresponding to additional machine requirements to meet the integrality condition of the original problem. The objective function value updated according to integer values gives an upper bound. Since the number of machines are rounded up, there exists at least one machine group with slack capacity. Therefore, in the next step we investigate the pairs of machine groups to check whether removing an operation, partially or entirely, from one machine group and reassigning it to another to reduce the cost. This may yield a reduction

- in the operating cost, if it is cheaper to process that operation in the second machine group,
- in the procurement cost, if the removal of that operation results in a saving of one machine,
- both in operating and procurement costs.

After all pairs of machine groups are investigated, the pair that achieves the maximum cost reduction is selected and the assignments and machine requirements are updated accordingly. Note here that capacity consumption of a certain operation may differ from one machine group to another since machines have different processing capabilities. Therefore, capacity consumption comparisons between machine groups are

performed accordingly, so are the machine requirements in each group updated. Table 2.2 depicts an example where 0.66 and 0.12 units of additional machines are needed in machine group *i* and machine group *j*, respectively (i.e. 1 machine is needed in each group). If a fraction of operation *j* is transferred at no additional procurement cost from machine group *l* to machine group *i* where the operating cost is lower then an improvement on the total cost is possible. In addition, if that fraction of operation *j* requires 0.12 units of machine capacity *l*, total cost is further improved by saving one machine. In that case, the additional machine requirement reduces to 0 and total procurement cost decreases by the cost of one machine of type *l*. Table 2.3 illustrates this case where 25% of operation *j* is removed from machine group *l* and reassigned to machine group *i*.

Table 2.2. Capacity allocation prior to operation exchange

	Operation *j*	..	Operation *k*	..	Total	Available machines	Machines required
Mach. Group *i*	0.18	..	0.86	..	2.66	2	1
...	..	..	..	..	..	..	..
Mach. Group *l*	0.82	..	0.00	..	3.12	3	1

Table 2.3. Capacity allocation after the operation exchange

	Operation *j*	..	Operation *k*	..	Total	Available machines	Machines required
Mach. Group *i*	0.43	..	0.86	..	2.82	2	1
...	..	..	..	..	..	..	..
Mach. Group *l*	0.57	..	0.00	..	3.00	3	0

At the end of Phase II, the final values of the additional machine requirements are obtained to build the new machine plan. In Phase III, we try to improve the lower bound using Lagrangian relaxation and/or Lagrangian decomposition method as described in Section 3. Finally, in Phase IV we solve a linear programming problem to find the optimal assignment scheme for the achieved machine plan and we update the upper bound on the optimal value of the objective function.

In what follows is a detailed description of cost reduction algorithm in Phase III. We first introduce the following notation for the data and variables employed in the algorithm:

q_i	number of additional type i machines required (may be decimal)
$Diff$	reduction in x_{ij}
MT_i	number of type i machines to be transferred
MN_i	number of total type i machines needed (may be decimal)
ME_k	excess capacity of type k machines

$$= \begin{cases} MA_k - MN_k , & \text{if } MA_k \geq MN_k \\ \text{floor}(MN_k) - MN_k , & \text{otherwise} \end{cases}$$

MA_i	number of available type i machines
TC	total operating and procurement costs
ΔTC	change in total cost
ΔOC_{ikj}	change in operational cost by transferring operation j, partially or entirely, from machine type i to machine type k

ΔTC_{ikj}	change in total cost by transferring operation j, partially or entirely, from machine type i to machine type k
$Diff_{BUY}$	reduction on x_{ij} if a new machine of type k is added
ΔTC_{BUY}	change in total cost if a new machine of type k is added
$Diff_{NOTBUY}$	reduction on x_{ij} if no machine is added to machine group k
ΔTC_{NOTBUY}	change in total cost if no machine is added to machine group k
ceil(x)	smallest integer number greater than or equal to x
floor(x)	greatest integer number less than or equal to x

Greedy Heuristic:

Step 0: Round up q_i obtained in LP1 to integer values to obtain Q_i : Q_i = ceil(q_i).

Update the objective function value TC.

Step 1: Let $I^* = \{ i \in I \setminus Q_i > 0 \}$. If $I^* = \varnothing$, go to Phase IV.

For $\forall i \in I^*$, go to Step 2.

Step 2: Set $MT_i = MN_i$ - floor(MN_i). If $MT_i = 0$, let $MT_i = 1$.

Go to Step 3.

Step 3: Let $J^* = \{ j \in J \setminus x_{ij} > 0 \}$

For $\forall k \in I - \{i\}$ and $\forall j \in J^*$ repeat Step 4-6.

Step 4: Set $\Delta x_{ij} = MT_i / a_{ij}$ and $\Delta VC_{ikj} = c_{kj} - c_{ij}$.

Go to Step 5.

Step 5: If $(\Delta x_{ij} \le x_{ij})$ set Drop = 1. Otherwise, set Drop = 0 and $\Delta x_{ij} = x_{ij}$.

Compute ME_k.

Step 6: If $(\Delta x_{ij} \le ME_k / a_{kj})$, let $Diff_{ikj} = \Delta x_{ij}$, and

$$\Delta TC_{ikj} = (\Delta VC_{ikj} * Diff_{ikj}) - (F_i * \text{Drop})$$

Otherwise, let

$$Diff_{BUY} = \Delta x_{ij}, \quad \Delta TC_{BUY} = (\Delta VC_{ikj} * Diff_{BUY}) - (F_i * Drop) + F_k$$

$$Diff_{NOTBUY} = ME_k / a_{kj}, \quad \Delta TC_{NOTBUY} = (\Delta VC_{ikj} * Diff_{NOTBUY})$$

$$\Delta TC_{ikj} = \min \{ \Delta TC_{BUY}, \Delta TC_{NOTBUY} \}$$

$$\text{Let } Diff_{ikj} = \begin{cases} Delta_{BUY} \ , & \text{if } \Delta TC = \Delta TC_{BUY} \\ Delta_{NOTBUY} \ , & \text{otherwise} \end{cases}$$

Step 7: Find $i^*, j^*, k^* = \text{argmin } \Delta TC_{ikj}$.

Let $\Delta TC^* = \Delta TC_{i^*k^*j^*}$ and $Diff^* = Diff_{i^*k^*j^*}$.

Step 8: If $\Delta TC^* = 0$, stop, no improvement on TC is possible. Otherwise:

$$TC = TC + \Delta TC^*$$

$$x_{i^*j^*} = x_{i^*j^*} - Diff^*, \ x_{k^*j^*} = x_{k^*j^*} + Diff^*$$

$$MN_{i^*} = MN_{i^*} - Diff^*.\ a_{i^*j^*}, \ \ MN_{k^*} = MN_{k^*} - Diff^*.\ a_{k^*j^*}$$

If $(Diff^* = MT_{i^*} / a_{i^*j^*})\ q_{i^*} = q_{i^*} - 1$

If $(Diff^* > ME_{k^*} / a_{k^*j^*})\ q_{k^*} = q_{k^*} + 1$

Let $Q_i = \text{ceil}(q_i)$ and $Q_k = \text{ceil}(q_k)$. Go to Step 1.

2.5. Experimental Analysis

The Operation-Machine Type Assignment Methodology is coded in C programming language. CPLEX® optimization software was used to solve LP problems. A series of computational results was carried out on a PC with 300 MHz Pentium II processor using several sets of randomly generated problems and/or problems based on industry. Since the industry data do not include operating and procurement cost figures, those costs are generated randomly. Considering different sized problems, the performance is evaluated using various combinations of cost structures. The procurement cost coefficient F_i is generated from uniform distributions: $D_1(F_i) \sim U(10, 12)$, $D_2(F_i) \sim U(10, 15)$, $D_3(F_i) \sim U(10, 20)$, and $D_4(F_i) \sim U(10, 40)$. The operating cost c_{ij} is a percentage of the procurement cost and is drawn from uniform distributions: $D_1(c_{ij}) \sim U(0.1, 0.3)$, $D_2(c_{ij}) \sim U(0.1, 0.9)$, and $D_3(c_{ij}) \sim U(0.1, 1.9)$. In other words, the

Table 2.4. Summary of results

		Prob1	Prob2	Prob3	Prob4	Prob5	Prob6	Prob7[a]
	Best	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.01%
(Z*-LB)/LB	Average	0.20%	0.74%	1.09%	0.50%	0.39%	0.22%	1.28%
	Worst	4.45%	5.72%	10.60%	3.07%	2.85%	2.13%	7.25%
	Best	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	
(UB-Z*)/Z*	Average	0.54%	1.90%	1.74%	1.48%	1.34%	1.02%	
	Worst	7.44%	8.81%	9.54%	8.65%	7.13%	5.52%	

[a] Z* could not been found for Prob7, the results are based on (UB-LB)/LB

operating cost is on the average 20% of the procurement cost using $D_1(c_{ij})$, 50% using $D_1(c_{ij})$, and 100% using $D_1(c_{ij})$. We consider three capacity utilization ratios: 0.60, 0.75, and 0.90.

We tested the solution procedure on seven types of problems over a wide range of values for the number of machine groups and number of operations. Each problem type was solved five times with the same input structure to achieve a reasonable level of confidence to validate the results (4 x 3 x 3 = 36 input parameters, 36 x 5 = 180 runs for each problem type). A summary of the results is presented in Table 2.4. Optimal solution for problem type 7 cannot be obtained because of the size of the problem and computer memory limitation. For all other problems, the optimal solution was achieved at least once using Lagrangian relaxation/decomposition technique and the greedy heuristic. The gap between the optimal solution and lower bound is less than 1.1% on the average in all problem instances and 10.6% in the worst case; and the gap between the upper bound and optimal solution is less than 1.9% on the average in all problem instances and 9.54% in the worst case.

The results show that better lower bounds and upper bounds are achieved when the number of operations in the problem is greater than the number of machine groups. Figure 2.2 and 2.3 illustrate the variation in mean gaps with respect to procurement costs and operating costs, respectively, at different utilization ratios. A pattern cannot be apparently observed between those parameters. On the other hand, although the gap

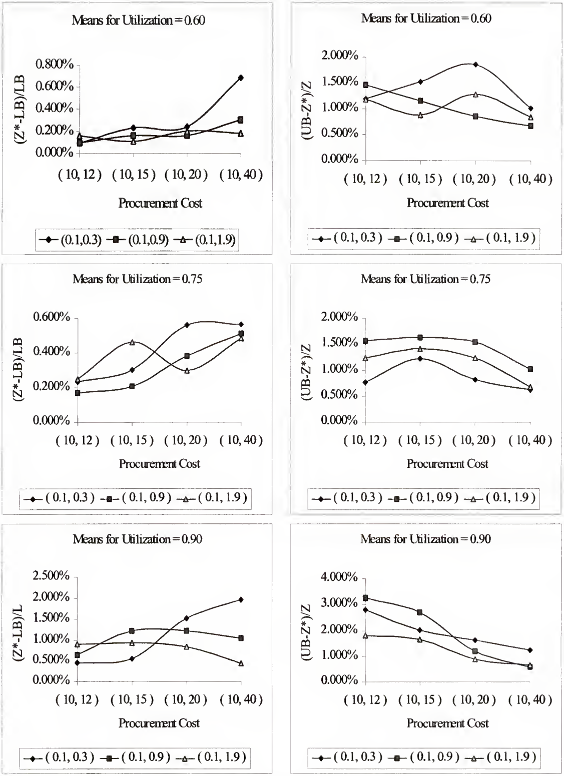


Figure 2.2. Mean gaps over procurement costs for different levels of utilization

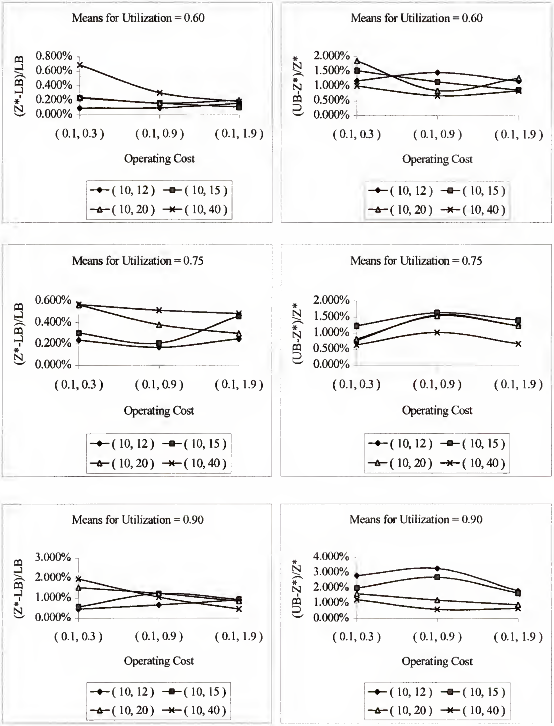


Figure 2.3. Mean gaps over operating costs for different levels of utilization

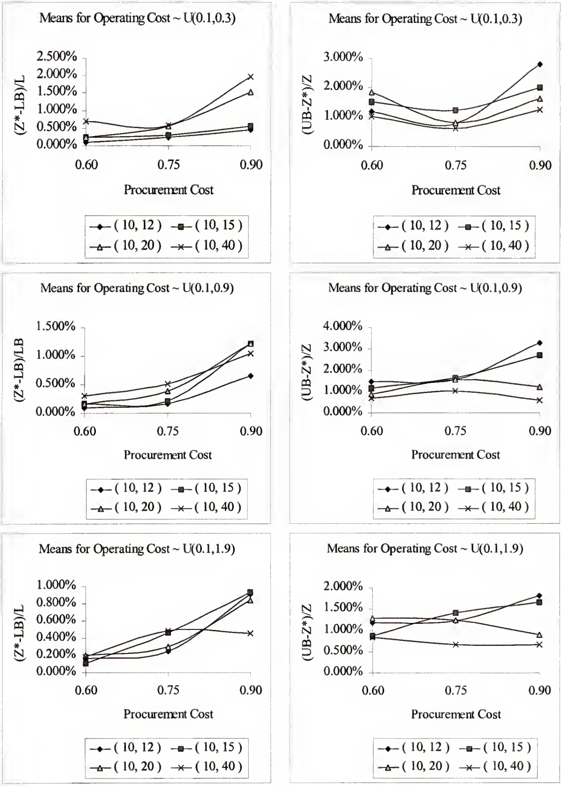


Figure 2.4. Mean gaps over utilizations for different procurement and operating costs

between the optimal solution and lower bound seems to increase as utilization increases we cannot make any statement for the gap between the upper bound and optimal solution. Figure 2.4 shows the change in mean gaps with respect to utilization ratios. Detailed results of all seven problems are given in Appendix A.

CHAPTER 3
MULTI-PERIOD CAPACITY ALLOCATION WITH MACHINE DUPLICATION IN SEMICONDUCTOR MANUFACTURING

3.1. Introduction

Most of the recent research in semiconductor manufacturing has focused on demand satisfaction, maximization of equipment utilization, minimization of production costs, and maximization of throughput with some capacity constraints. Linear Programming is often proposed as a tool for production planning and scheduling in semiconductor manufacturing. However, LP formulations can be very large for large organizations with complex production environments such as semiconductor industry requiring a very long time to generate the input data files that need to be fed into mathematical planning software and huge amount of memory and disk space to store these data. Golovin (1986) points out the difficulty of selecting an appropriate objective function in the planning of the semiconductor manufacturing and shows that a detailed formulation of the problem in an integrated manner based upon a mathematical programming model is intractable and requires data which cannot be obtained reliably. He proposes a hierarchical approach such as that suggested by Bitran et al. (1981, 1982).

Leachman (1993) gives an optimization-based corporate-level production planning system that includes multiple facilities and treats entire production processes in each facility as integral entities. His model generates capacity-feasible start and out

schedules for each manufacturing facility in the company. Leachman and Carmon (1992) analyze capacity of semiconductor production facilities in which manufacturing operations may be performed by alternative machine types. This type of operations can be performed by different machine types with different processing times. They propose a modeling technique that greatly reduces the size of the LP problems. Hung and Wang (1997) present an alternative model for formulating the alternative machine capacitated planning problem as an extension of the technique proposed by Leachman and Carmon (1992). Their technique considerably reduces the size of the LP problem for bin

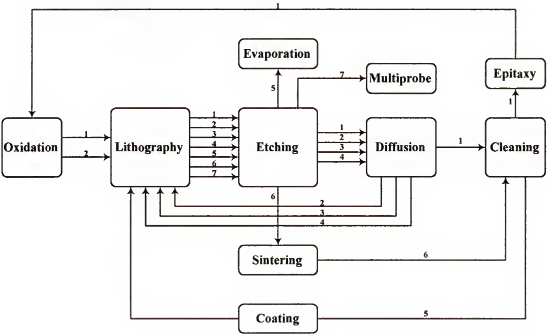


Figure 3.1. Simple silicon TTL Integrated Circuit process flowchart

allocation planning in semiconductor manufacturing, thus saving substantial amount of computer resources required. The interested reader is referred to Uzsoy *et al.* (1992) for a comprehensive review of the research in the semiconductor production planning and scheduling.

The alternative machine type capacity allocation problem with duplicated machines in a complex semiconductor manufacturing environment is a difficult task. Machines have different manufacturing characteristics according to the process and product involved. While newer machines are normally capable of processing advanced products as well as older products, older machines can only process older products, may require longer processing times and may have lower yields and lower utilization levels due to increasing maintenance requirements. In this chapter, we address this problem by recognizing capacity limitations of the individual machines as well as reducing operating and investment costs related to the machines and inventory holding costs. We investigate the tradeoff between allocating a constant capacity for the planning horizon and holding inventory and propose a Lagrangian relaxation based heuristic approach to solve the problem.

The remainder of this chapter is organized as follows. In Section 3.2, we describe the problem and formulate a mathematical model. In Section 3.3, we present a Lagrangian-based solution procedure to obtain lower bound and a good feasible solution to the problem. Section 3.4 contains experimental analysis of the proposed solution procedure on randomly generated data.

3.2. Problem Statement and Formulation

Manufacturing a computer chip is a complex process involving hundreds of steps and requiring from a few days to up to three months of processing time to complete. The semiconductor manufacturing process consists of various operation types such as oxidation, deposition, lithography, etching, ion implantation, etc. performed on different batches of products (wafers). Some of these operation types are repeated several times to build different layers on top of the wafer and wafers may require up to 400 operation steps at a number of work centers. Each wafer type follows a particular sequence throughout its fabrication process visiting different work centers. Specifically, each wafer makes multiple visits to the same work center at different points in the fabrication process. Such a product flow is known as a reentrant product flow and is illustrated in Figure 3.1. The common practice in the layout of work centers for different types of processes is to group similar operation types together: furnaces are grouped in one area, ion implanters in another, and so forth. This layout requires wafers moving back and forth between work centers but it allows the utilization of the same equipment to process wafers at different steps. For instance, all oxidation processes may be completed on the same machine or in the same work center although the wafers are required to travel to other work centers between these oxidation process steps.

Since the specifications of the process performed on the wafer differ from one visit to another, each visit is referred to as a distinct step and each processing step is given a specific name. This name indicates both the wafer type/s and the process to be performed on the wafer type/s. Throughout this chapter wafer type/s at a particular

processing step is/are referred to as an "operation." In this environment, we focus on the loading of operations to alternative machine types where machine duplication is allowed for certain machine groups. We consider a population of machines that are capable of processing these operations only. Machines having same characteristics are gathered into machine groups and each machine group represents a set of parallel identical machines with the same processing capabilities. In this context, machines groups are classified as primary machine groups and secondary machine groups where primary machines are the most efficient machines to process the specified operations and secondary machines are the alternative machines in case additional machine capacity is needed. Only primary machines may be procured if further capacity is required to meet the production schedules.

The problem of modeling the capacity of alternative, non-identical resources arises in almost all manufacturing environments where process technology is evolving. Suprisingly, published research on this topic is scant (Leachman and Carmon, 1992). In this section, we develop a mixed-integer programming formulation to minimize operating cost of machines, procurement cost of primary machines, and inventory holding cost. For each operation, there are allocation variables to spread the product-operation production quantities among alternative machine groups. Inventory variables and integer variables for additional machine requirements are used to meet the production volumes at each period. Inventory balance constraints are formulated to guarantee consistency of production volumes between consecutive periods and resource constraints are formulated

Table 3.1. Illustrative data for five operations processed on lithography machines

Machine Type	Quantity Available	Net Production Hrs/24 Hr	Utilization
PL - 100	5	18.0	0.75

Operation Number	Scheduled Volume	Production Capacity	% Yield	Number of machines needed
8	212	880	97.44	0.23
17	197	1198	96.38	0.16
24	25	476	91.27	0.05
28	52	549	93.44	0.09
39	155	1020	95.71	0.15

both to satisfy capacity requirements and to determine additional machine requirements. Given the number of machines in each machine group and capacity consumption of each operation type in each machine group, our objective is to find the optimal assignment scheme of operations to machine groups. To the best of our knowledge no research exists in the literature that addresses this type of problem with integer and continuous variables involving both assignment and capacity decisions.

Table 3.1 illustrates sample data for a machine group consisting of ion implanters and four operations that can be processed on this machine group. For simplicity of exposition, we will assume 100% yields and zero initial inventories. We introduce the following notation for the data concerning our capacity allocation problem:

$t \in T$ is the time period index, where $t = 1, \ldots, r$,

$i \in I$ is the machine group index, where $i = 1, \ldots, m$,

$j \in J$ is the operation index, where $j = 1, \ldots, n$,

N_{ti} denotes the number of machines currently available in each machine group i at period t,

$[P]$ is the capability matrix: the binary entry p_{ij} is 1 if machine group i can perform operation j and 0 otherwise,

V_{tj} is the daily scheduled production volume of operation j at period t,

Y_{tij} is the yield of machine type i with respect to operation j at period t,

PC_{tij} is the number of wafers of type j that machine group i can process daily at period t,

U_{ti} denotes the utilization of the machine group i at period t,

(= Net production hours / 24 hours)

β_{tij} is the fraction of machine type i required to perform each operation j,

(= $1 / (PC_{tij} * Y_{tij})$ if p_{ij} is equal to 1)

c_{tij} discounted operating cost of each machine of type i to process operation j at period t,

F_{ti} discounted capitalized procurement cost of an additional machine of type i at period t,

h_{tj} discounted cost of holding one unit of inventory of wafer type j at period t.

Before we present the problem formulation we define the following production, inventory, and machine requirement variables:

X_{tij} is the number of units of wafer type j assigned to machine group i at period t,

I_{tj} is the number of units of wafer type j held in inventory at period t,

Q_{ti} is the number of additional machine of type i needed at period t.

A comprehensive mathematical formulation of the multi-period alternative machine type capacity allocation problem with duplicated machines may be expressed as follows:

Problem MPCAP

$$\text{Min} \quad \sum_{t\in T}\sum_{i\in I}\sum_{j\in J} c_{tij}\,\beta_{tij}\,X_{tij} + \sum_{t\in T}\sum_{j\in J} h_{tj}\,I_{tj} + \sum_{t\in T}\sum_{i\in P} F_{ti}\,Q_{ti} \tag{1}$$

$$\text{s.t.} \quad I_{(t-1)j} + \sum_{i\in I} X_{tij} - I_{tj} = V_{tj}\,, \qquad \forall t\in T,\ \forall j\in J \tag{2}$$

$$\sum_{j\in J} \beta_{tij} X_{tij} \le N_{ti} U_{ti}\,, \qquad \forall t\in T,\ \forall i\in S \tag{3}$$

$$\sum_{j\in J} \beta_{tij} X_{tij} \le N_{ti} U_{ti} + \sum_{\tau=1}^{t} Q_{\tau i} U_{\tau i}\,, \qquad \forall t\in T,\ \forall i\in P \tag{4}$$

$$Q_{ti} \ge 0 \text{ and integer}\,, \qquad \forall t\in T, \forall i\in I \tag{5}$$

$$0 \le X_{tij} \le Mp_{ij}\,, \qquad \forall t\in T, \forall i\in I, \forall j\in J \tag{6}$$

For a given U_{ti} we define $b_{tij} = \beta_{tij} / U_{ti}$ and replace constraints (3) and (4) with:

$$\sum_{j \in J} b_{tij} X_{tij} \leq N_{ti} \,, \qquad \forall t \in T, \ \forall i \in S \qquad (7)$$

$$\sum_{j \in J} b_{tij} X_{tij} \leq N_{ti} + \sum_{\tau=1}^{t} Q_{\tau i} \,, \qquad \forall t \in T, \ \forall i \in P \qquad (8)$$

The objective function of the formulation is to minimize operating cost of machines, procurement cost of additional machines, and inventory holding cost. Constraints (2) ensure that the demands at each period are satisfied. Constraints (7) express the capacity limits on the "secondary" machine groups and constraints (8) express the capacity limits on the "primary" machine groups where capacity may be increased by adding new machines. We prefer to express machine capacity restrictions under two sets of constraints to emphasize two different classes of machine groups. These two constraints may as well be formulated as one constraint set similar to (8). Constraints (5) impose integrality and non-negativity restrictions on the additional machine variables. Constraints (6) express the usual non-negativity conditions on assignment variables and ensure that operations are assigned to machine groups that are capable of processing them. M is a sufficiently large number.

In the next section, we discuss a Lagrangian relaxation based solution procedure to obtain a good lower bound on the minimum value of problem *MPCAP* and a feasible upper bound.

3.3. Description of the Lagrangian-based Heuristic Solution Method

Lagrangian relaxation has been successfully employed by numerous researchers in the past in many integer/mixed integer programming applications. The approach is based on the observation that many difficult integer programming problems can be viewed as they are composed of two types of constraints: “nice” constraints and “complicating” constraints. Lagrangian relaxation is formed by multiplying the complicating constraints with corresponding penalty costs (Lagrangian multipliers) and including them into the objective function. By dualizing those constraints, we obtain a problem that is easy to solve and whose optimal value gives a lower bound on the optimal value of the minimization problem. Although the method does not guarantee that a solution will be found for all problems, it is often worth trying since it is so simple

Step 1	Initialize Lagrangian multipliers.
Step 2	Update the lower bound on the minimum value of the objective function by solving relaxed model.
Step 3	Compute the machine groups' workloads and inventories from the current solution and update Lagrangian multipliers with respect to violations of the constraints.
Step 4	Update the upper bound on the minimum value of the objective function; if a stopping criteria is met, stop; otherwise go to Step2.

Figure 3.2. Outline of the heuristic procedure

compared to other methods. The reader is referred to Geoffrion (1974) and Fisher (1981) for a detailed study on the Lagrangian relaxation technique.

Our solution method is based on the Lagrangian relaxation of the capacity constraints (7), (8), and a set of cumulative machine requirements constraints, which is discussed in the next section. An overview of the overall solution procedure for problem *MPCAP* is presented in Figure 3.2. Through Lagrangian relaxation of the above mentioned constraints in Step 2 we obtain two subproblems, one with assignment and inventory variables, the other with additional machine requirement variables. The first problem further decomposes into several subproblems that are easily solved by inspection, as is the latter. The solutions of these problems are used to compute a lower bound on the minimum value of the objective function.

In Step 3, the assignment scheme and inventory levels resulting from Step 1 are determined and the resource workloads are computed using equations (2) – (4). The violations in the constraints are employed to update the Lagrangian multipliers applying a subgradient optimization technique. Finally, the upper bound on the minimum value of the objective function is computed in Step 4. The violations of capacity constraints (3) are eliminated through an operation shifting procedure and then a machine reduction procedure is applied to equations (4).

3.3.1. Computation of the Lower Bound

Before we proceed with the Lagrangian relaxation of the problem *MPCAP* we first consider adding a set of valid inequalities as lower bounds on the total number of

additional machines needed to satisfy the cumulative demand from periods 1 through t, where $t = 1, \ldots, r$:

$$\sum_{i \in P} \sum_{\tau=1}^{t} \sum_{z=1}^{\tau} Q_{zi} \geq K_t, \qquad \forall t \in T \qquad (9)$$

These constraints may provide better bounds in the relaxed problem. K_t are the minimum cumulative additional machine requirements to meet the demands of periods 1 through t and are best obtained by solving the following problem:

Problem MPCMR

$$\text{Min} \quad \sum_{t \in T} \sum_{i \in P} Q_{ti}$$

$$\text{s.t.} \quad \sum_{\tau=1}^{t} \sum_{i \in I} X_{\tau ij} \geq \sum_{\tau=1}^{t} V_{\tau j}, \qquad \forall t \in T, \forall j \in J$$

and (5) - (8)

and then setting

$$K_t = \sum_{i \in P} \sum_{\tau=1}^{t} \sum_{z=1}^{\tau} Q^*_{zi}, \quad t = 1, \ldots, T$$

It is as difficult to solve problem *MPCMR* as problem *MPCAP*. Therefore we propose the following set of uncapacitated problems that can be easily solved by inspection as an alternative procedure to obtain K_t.

Problem CMR_t

$$\text{Min} \quad \sum_{\tau=1}^{t}\sum_{i\in I}\sum_{j\in J} b_{\tau ij} X_{\tau ij}$$

$$\text{s.t.} \quad \sum_{\tau=1}^{t}\sum_{i\in I} X_{\tau ij} \geq \sum_{\tau=1}^{t} V_{\tau j}, \qquad \forall j \in J$$

$$X_{\tau ij} \geq 0, \qquad \forall i \in I, \forall j \in J$$

$$\text{where } K_t = \left\lceil \sum_{\tau=1}^{t}\sum_{i\in I}\sum_{j\in J} b_{\tau ij} X^*_{\tau ij} - N_{\tau i} \right\rceil$$

The solution procedure for Problem CMR_t, $t = 1, \ldots, r$ is as follows: for each period $\tau = 1, \ldots, t$ assign the whole volume of operation j to the fastest machine group i where it consumes least amount of resources regardless of the machine group's capacity but subject to its capability. Identify the number of additional machines required in each machine group i. Round up the difference between the total number of machines needed and total number of machines currently available.

Proposition: Constraints (9) and $\sum_{i\in P}\sum_{\tau=1}^{t}(t-\tau+1)Q_{\tau i} \geq K_t, \qquad \forall t \in T \qquad (10)$

are equivalent.

Proof: $\sum_{i\in P}\sum_{\tau=1}^{t}\sum_{z=1}^{\tau} Q_{zi} = \sum_{\substack{i\in P\\ t\geq 1}} Q_{1i} + \sum_{\substack{i\in P\\ t\geq 2}} (Q_{1i} + Q_{2i}) + \ldots + \sum_{\substack{i\in P\\ t=r}} (Q_{1i} + Q_{2i} + \ldots + Q_{ti})$

$$= \sum_{\substack{i \in P \\ t \geq 1}} tQ_{1i} + \sum_{\substack{i \in P \\ t \geq 2}} (t-1)Q_{2i} + \ldots + \sum_{\substack{i \in P \\ t = r}} Q_{ti}$$

$$= \sum_{i \in P} \sum_{\tau=1}^{t} (t - \tau + 1)Q_{\tau i}$$

Our solution method is based on the Lagrangian relaxation of the capacity constraints (7), (8), and cumulative machine lower bounding constraints (10). The overall solution procedure for problem *MPCAP* is outlined in Figure 3.2. The formulation of the relaxed problem is as follows:

Problem $LR(\lambda,\mu)$

$$\text{Min} \quad \sum_{t \in T} \sum_{i \in I} \sum_{j \in J} \left(c_{tij}\beta_{tij} + \lambda_{ti} b_{tij}\right) X_{tij} + \sum_{t \in T} \sum_{j \in J} h_{tj} I_{tj} + \sum_{t \in T} \sum_{i \in P} F_{ti} Q_{ti}$$

$$- \sum_{t \in T} \sum_{i \in P} \sum_{\tau=1}^{t} \lambda_{ti} Q_{\tau i} - \sum_{t \in T} \sum_{i \in P} \sum_{\tau=1}^{t} \mu_t (t - \tau + 1) Q_{\tau i} + C \qquad (11)$$

s.t. (2), (5), and (6)

where $\lambda,\mu \geq 0$ and C is constant:

$$C = -\sum_{t \in T} \sum_{i \in I} \lambda_{ti} N_{ti} + \sum_{t \in T} \mu_t K_t$$

The term involving Lagrangian multiplier λ_{ti} and Q_{ti} in the objective function can alternatively be expressed as follows:

$$\sum_{t\in T}\sum_{i\in P}\sum_{\tau=1}^{t}\lambda_{ti}Q_{\tau i}$$

$$= \sum_{i\in P}\left[\lambda_{1i}Q_{1i} + \lambda_{2i}(Q_{1i} + Q_{2i}) + \ldots + \lambda_{ri}(Q_{1i} + Q_{2i} + \ldots + Q_{ri})\right]$$

$$= \sum_{i\in P}\left[(\lambda_{1i} + \lambda_{2i} + \ldots + \lambda_{ri})Q_{1i} + (\lambda_{2i} + \ldots + \lambda_{ri})Q_{2i} + \ldots + \lambda_{ri}Q_{ri}\right]$$

$$= \sum_{i\in P}\sum_{\tau=1}^{r}\lambda_{\tau i}Q_{1i} + \sum_{i\in P}\sum_{\tau=2}^{r}\lambda_{\tau i}Q_{2i} + \ldots + \sum_{i\in P}\sum_{\tau=r}\lambda_{ri}Q_{ri}$$

$$= \sum_{t\in T}\sum_{i\in P}\sum_{\tau=t}^{r}\lambda_{\tau i}Q_{ti}$$

Using a similar expression for the term with the Lagrangian multiplier μ the objective function is reformulated as follows:

$$\sum_{t\in T}\sum_{i\in I}\sum_{j\in J}(c_{tij}\beta_{tij} + \lambda_{ti}b_{tij})X_{tij} + \sum_{t\in T}\sum_{j\in J}h_{tj}I_{tj}$$

$$-\sum_{t\in T}\sum_{i\in P}\left[F_{ti} - \sum_{\tau=t}^{r}(\lambda_{\tau i} + (\tau - t + 1)\mu_{\tau})\right]Q_{ti} + C \qquad (12)$$

Lagrangian problem $LR(\lambda,\mu)$ nicely decomposes into two subproblems, one involving the X_{tij} and I_{tj} variables only and the other involving Q_{ti} variables only. First problem further decomposes into n uncapacitated multi-period single-item production planning problem which can be easily solved by inspection. A procedure to solve this problem is as follows:

set all X_{tij}, I_{tj} *to zero*

for $t = 1, \ldots, r$ *do*

$\quad i^* = argmin_{i \epsilon I} \{c_{tij}\, \beta_{tij} + \lambda_{ti} b_{tij}\}$

$\quad$ *set* $X_{ti^*j} = V_{tj}$

endfor

for $t = r, \ldots, 2$ *do*

$\quad$ *for* $\tau = (t\text{-}1), \ldots, 1$ *do*

$\quad\quad$ *if* $\{(c_{ti^*j}\, \beta_{ti^*j} + \lambda_{ti^*} b_{ti^*j} > c_{\tau i^*j}\, \beta_{\tau i^*j} + \lambda_{\tau i^*} b_{\tau i^*j} + h_{\tau j}) \;\&\; (X_{\tau i^*j} > 0)\}$ *forbreak*

$\quad\quad$ *else* $\tau = \tau - 1$

$\quad$ *endfor*

$\quad$ *if* $(\tau > 0)$ *do*

$\quad\quad$ *for* $z = \tau, \ldots, t\text{-}1$ *do* $I_{\tau j} = I_{\tau j} + X_{ti^*j}$

$\quad\quad X_{\tau i^*j} = X_{\tau i^*j} + X_{ti^*j}$

$\quad\quad X_{ti^*j} = 0$

$\quad\quad I_{tj} = I_{tj} - V_{tj}$

$\quad$ *endif*

endfor

Second problem can also be solved by inspection: if the coefficient of the integer variable Q_{ti} is positive then Q_{ti} is set to zero. Otherwise, Q_{ti} would be infinite. We impose an upper bound for Q_{ti} to avoid an unbounded solution.

We use a subgradient optimization procedure to update the Lagrangian multipliers. Given initial values of λ^0 and μ^0, we generate a sequence of Lagrangian multipliers λ^k and μ^k through the addition of a direction vector which is multiplied by a step size θ^k where θ^k is a positive scalar. Hence, the updating of Lagrangian multipliers of the constraint sets (7) and (8) is done according to the equations

$$\lambda_{ti}^{k} = max\left\{0; \lambda_{ti}^{k-1} + \theta^k \left(\sum_{j \in J} b_{tij} X_{tij} - N_{ti} \right)\right\}, \qquad \forall t \in T, \forall i \in S$$

$$\lambda_{ti}^{k} = max\left\{0; \lambda_{ti}^{k-1} + \theta^k \left(\sum_{j \in J} b_{tij} X_{tij} - N_{ti} - \sum_{\tau=1}^{t} Q_{\tau i} \right)\right\}, \quad \forall t \in T, \forall i \in P$$

and Lagrangian multipliers of the constraints (10) is updated according to the equation

$$\mu_t^k = max\left\{ 0 ; \mu_t^{k-1} + \theta^k \left(K_t - \sum_{i \in P} \sum_{\tau=1}^{t} (t - \tau + 1) Q_{\tau i} \right)\right\}, \quad \forall t \in T$$

The step size θ^k is updated at each iteration k using the following equation:

$$\theta^k = \delta^k \frac{(\mathrm{UB} - \mathrm{LB}(\lambda^{k-1}, \mu^{k-1}))}{\sqrt{\begin{array}{l} \displaystyle\sum_{t \in T} \sum_{i \in S} \left(\sum_{j \in J} b_{tij} X_{tij} - N_{ti} \right)^2 + \\ \displaystyle\sum_{t \in T} \sum_{i \in P} \left(\sum_{j \in J} b_{tij} X_{tij} - N_{ti} - \sum_{\tau=1}^{t} Q_{\tau i} \right)^2 + \\ \displaystyle\sum_{t \in T} \left(K_t - \sum_{i \in P} \sum_{\tau=1}^{t} (t - \tau + 1) Q_{\tau i} \right)^2 \end{array}}}$$

The step size θ^k depends on the parameter δ^k ($0 < \delta^t \leq 0.025$), on the gap between the current lower bound LB(λ^{k-1}, μ^{k-1}) and the estimated minimum value of the objective function of the relaxed problem, which is approximated by the upper bound UB obtained by applying a heuristic method, and on the Euclidean norm of the deviations in the relaxed constraints (7), (8), and (10). The sequence δ^k is determined by setting $\delta^0 = 0.025$ and by dividing δ^k by 1.5 whenever LB(λ^{k-1}, μ^{k-1}) does not increase after a fixed number of iterations. We terminate this procedure when one of the following stopping criteria is met:

1. An iteration number limit,
2. Maximum gap between the lower and upper bounds,
3. A limit on the value of the Euclidean norm of the deviations.

The Lagrangian problem $LR(\lambda,\mu)$ has the Integrality Property, that is the optimal solution does not change if we drop the integrality restriction on Q_{ti}. Thus, Lagrangian relaxation cannot do better than LP relaxation. However, we find promising to use Lagrangian relaxation since the problems we consider are of very large scale and Lagrangian relaxation can provide good lower bounds substantially faster than the standard LP relaxation.

3.3.2. Computation of the Upper Bound

At each iteration of the Lagrangian relaxation the upper bound is computed by first modifying the solution to obtain feasibility and then by applying a procedure to

improve the bound. Since the Lagrangian relaxation results in a solution that is feasible with respect to the inventory balance constraints, we must ensure that the production quantities do not lead to over usage of the secondary machines. With respect to the resource constraints, a feasible production plan can easily be created by shifting operations from overloaded secondary machine groups to capacity flexible primary machine groups. The selection of which operations to be shifted is done by sorting the machine group-operation pairs at each period in non-increasing order of the product of consumption and operating cost coefficients and then choosing the candidate machine group-operation pair following that sequence. Note here that since another secondary machine group may as well be under utilized in the original plan and therefore may be included in the candidate list. We continue performing this shifting procedure until the machine requirements in all overloaded secondary machine groups at all periods are at their respective capacity levels. An operation quantity may be partially reassigned if it is not necessary to shift the whole production volume to bring the total resource consumption in certain machine group to capacity. The procedure is as follows:

for $t = 1, \ldots, r$ *sort* $OperCost_{tij}{*}Consumption_{tij}$ *in non-increasing order*

for $t = 1, \ldots, r$ *do*

  for $i = 1, \ldots, m_1$ *do*

    while $(MachReqd_{ti} > AvailMach_{ti})$ *do*

      shift X_{tj} *from* $MachGroup_i$ *to* $MachGroup_{i^*}$ *following the sequence*

      update $MachReqd_{ti}$ *and* $MachReqd_{ti^*}$

endwhile

endfor

endfor

for $t = 1, \ldots, r$ *do*

for $i = m_1, \ldots, m$ *do*

while ($MachReqd_{ti} > AvailMach_{ti} + MachUB_{ti}$) *do*

shift X_{tj} *from* $MachGroup_i$ *to* $MachGroup_{i^*}$ *at period* t *or* $t\text{-}1$ *following the sequence*

update $MachReqd_{ti}$ *and* $MachReqd_{ti^*}$ *or* $MachReqd_{(t-1)i^*}$

endwhile

endfor

endfor

MachUB is the upper bound on the number of additional machines imposed to avoid unbounded solution and m_1 is the number of secondary machine groups.

After a feasible production plan is constructed, in the second stage we try to improve the solution by means of operation transfers between primary machine groups within the same period or different periods. An operation can be transferred from a period to an earlier period only in order to not lead to any backorders. The procedure is performed in a similar fashion using the above mentioned sequence of machine group-operation pairs starting with the current period and then proceeding with the preceding periods in an attempt to avoid inventories.

3.4. Experimental Analysis

The heuristic solution procedure is coded in C programming language. A series of computational results is carried out on a PC with 300 MHz Pentium II processor using several sets of randomly generated problems of differing sizes. The performance of the procedure is evaluated using various combinations of cost, utilization, and primary/secondary machine groups structures. Procurement cost coefficient F_{ti}, operating cost coefficient c_{tij}, and utilization U_{ti} are drawn from uniform distributions using the parameters shown in Table 3.2. The operating cost is obtained as a percentage of the procurement cost, i.e. operating cost is on the average 20% of the procurement cost using $U(0.1, 0.3)$, 40% of the procurement cost using $U(0.1, 0.7)$, etc. F_{ti} and c_{tij} are generated for the first period and discounted for succeeding periods. Similarly, U_{tij} are generated

Table 3.2. Experimental design

Parameters	Values used	Total
Procurement cost	(10, 11), (10, 14), (10, 17), (10, 20)	4
Operating cost[a]	(0.1, 0.3), (0.1, 0.7), (0.1, 1.1), (0.1, 1.5), (0.1, 1.9)	5
Utilization	(0.65, 0,75), (0.65, 0.85), (0.65, 0.95)	3
Number of periods	20, 40	2
Number of machine groups/operations	10/200 15/300 20/400 25/500 30/600	
Machine group structures[b]		29
Total number of problems		3480

[a] Variable cost is a fraction of the procurement cost
[b] 5 for 10/200 and 15/300, 6 for 20/400 and 25/500, and 7 for 30/600

for the first period and then decreased using a certain constant ratio for the following periods considering decline in the utilization of machines over time. Production volumes and production capacities are drawn from uniform distributions $U(1, 400)$ and $U(50, 1500)$, respectively. The number of process steps repeated throughout the manufacturing of the wafer is also generated form a uniform distribution $U(3, 10)$.

We tested our procedure on ten problem types over a wide range of values of number of machine groups and operations. Each problem type is solved for different primary/secondary machine group structures with the same input data. The smallest problem we consider consists of 44,340 constraints, 44,060 variables, of which 60 are integers and the largest problem has 747,160 constraints, 744,960 variables, of which 960 are integers. A summary of the results is presented in Table 3.3. The gap between the upper bound and lower bound is computed as (UB-LB)/UB. Although the worst case performances do not seem promising in terms of absolute values, we were able obtain fairly good solutions relatively rapidly for large problems. Note here that it took more than 80 minutes to obtain even the LP optimal solution in some problem instances of 40 periods, 500 operations, and 25 machine groups that we tested using the CPLEX® optimization software. The average CPU time of all 360 problem instances of this structure using our solution procedure is 337 seconds and the maximum CPU time is 520 seconds.

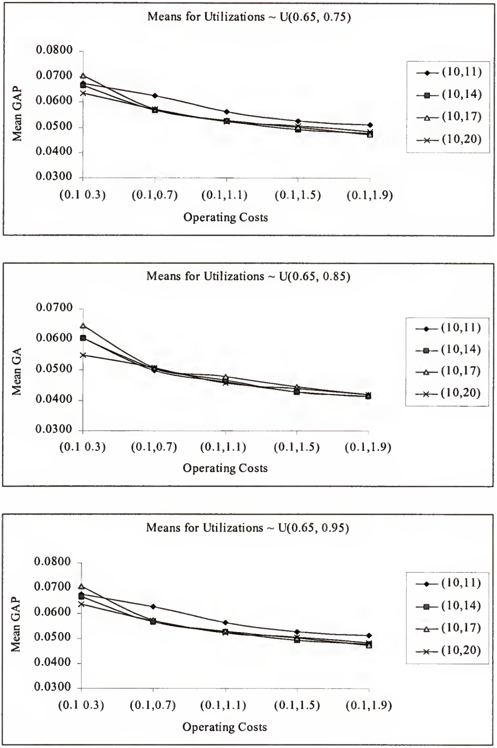


Figure 3.3. Mean gaps over operating costs for different levels of utilization

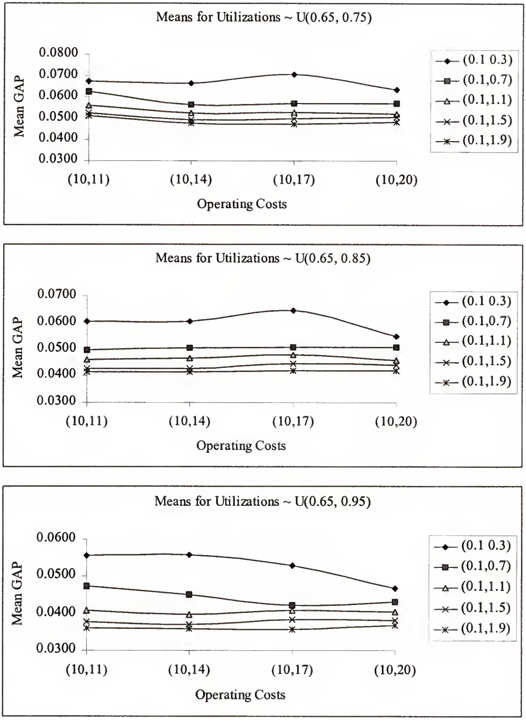


Figure 3.4. Mean gaps over procurement costs for different levels of utilization

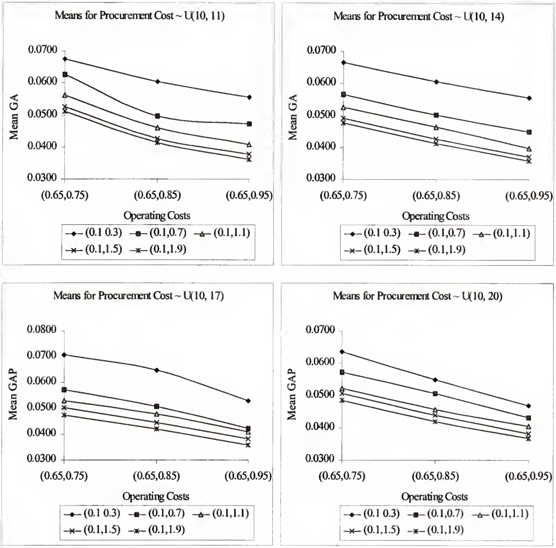


Figure 3.5. Mean gaps over utilizations for different procurement and operating costs

Figure 3.3 illustrates the change in the mean gaps with respect to operating costs for different utilization distributions. As the mean of the distribution of operating cost increases, the mean gap decreases. On the other hand, no significant variation is observed in the mean gap with respect to the procurement costs (Refer to Figure 3.4). Figure 3.5

Table 3.3. Summary of results showing the gaps between upper and lower bounds

		Machine Groups/Operations				
		10/200	15/300	20/400	25/500	30/600
	Best	0.0538	0.0320	0.0217	0.0193	0.0271
20 Periods	Average	0.0809	0.0565	0.0499	0.0459	0.0515
	Worst	0.1163	0.0903	0.0975	0.0935	0.0960
	Best	0.0173	0.0194	0.0326	0.0258	0.0176
40 Periods	Average	0.0541	0.0319	0.0487	0.0406	0.0327
	Worst	0.1045	0.0573	0.0958	0.0707	0.0734

exhibits the change in the mean gaps with respect to the distribution of utilization. We can observe that the solution procedure performs better as the mean utilization increases. This is due to the fact that more additional machines are needed if the utilization is smaller. Detailed results for differing problem structures are included in Appendix B.

CHAPTER 4
INTEGRATING PRINTED CIRCUIT BOARD SCHEDULING AND COMPONENT GROUPING IN AN OPENSHOP MANUFACTURING ENVIRONMENT

4.1. Introduction

Printed circuit board (PCB) assembly consists of a PCB and various electrical and electronic components that are placed onto the board. There is a tremendous variation in the technology of circuit boards and the components. The interested reader is referred to Kear (1987) and McGinnis et al. (1992) for a detailed description of printed circuit assembly technology.

Modern PCB production typically uses computer controlled insertion machines to automatically assemble electronic components onto the boards in a flow shop type of production line. Production control of this assembly process deals with several problems such as the selection of the component types to allocate to each machine, the determination of the sequence of PCBs for production, the evaluation of the arrangement of feeders onto the machine, and the determination of the sequence of the placement operations. A comprehensive review of the literature in the PCB assembly research area may be found in McGinnis et al. (1992) and Askin et al. (1994).

We discuss the component family formation and scheduling problem in three stages. The first stage deals with the problem of component grouping in order to minimize the number of machines visited. The second stage loads component families to

automatic placement machines and attempts to balance the workload. The third stage addresses the scheduling of the PCBs to machines in order to minimize the maximum completion time (makespan) with the secondary objective of reducing the mean flow time. Minimizing maximum completion time would either increase the production capacity or reduce work-in-process inventory and reducing the mean flow time would also reduce work-in-process inventory without adversely affecting production capacity. Thus, the issues addressed are

1. Grouping component types into families,
2. Loading component families to machines, and
3. Routing PCBs to machines.

In this study, an assembly system consisting of identical decoupled insertion machines is examined. The machines place electronic components onto PCBs using surface mount technology. Each PCB has a bill of components specifying the component types, the number of each type of component, and the feeder slots that each component requires on each machine. For the environment studied, insertion time is assumed to be a fixed (unitary) constant for each component regardless of PCB type and the component's location on the board. Thus, the total insertion time of all components on a PCB the sum of the total number of components it requires.

We also assume zero machine setup times when switching from one PCB type to another. This assumption is based on the fact that components are already loaded on each machine. To solve the allocation problem of the component types to machines, component families are formed explicitly considering the machine capacity constraint in

terms of feeder slots. Note here that feeder slots that each component type requires may be different and each component is assigned to a single machine. When a machine starts placing a component type on a board, no interruption is allowed until the whole operation is completed for that board (i.e., no preemption occurs). Since there are no precedence requirements on the operations performed PCBs can visit the machines in any order. Furthermore, if the machines are decoupled, then shop configuration would be an openshop.

The remainder of this chapter is organized as follows. Section 4.2 reviews the relevant literature to PCB manufacturing and presents an overview of the openshop scheduling. Section 4.3 describes the formulation of the clustering problem under consideration and discusses the algorithms developed for component family formation, component loading and PCB scheduling. Section 4.4 contains the experimental analysis of the proposed methodology on industry data.

4.2. Relevant Literature

Over the last two decades, many researchers have studied the assembly of printed circuit boards in an effort to develop models to support the process planning, production planning, and scheduling decisions. Typical objective in these decision models includes minimizing cycle time and/or setup time. Drezner and Nof (1984), Ball and Magazine (1988), Ahmadi, Grotzinger, and Johnson (1988), Francis, Hamacher, Lee, and Yeralan (1989), Grotzinger (1992) address the problem of assigning feeders to machines and sequencing placement activities to minimize the cycle time for each board type. These

papers also discuss issues such as the requirement to change nozzles for some component types, concurrent movement of pick and place activities, and multiple delivery heads.

Lofgren and McGinnis (1986), Carmon, Maimon, and Dar-El (1989), Maimon and Shtub (1991) consider clustering techniques for components and boards to reduce the setup time. Lofgren and McGinnis (1986) group similar component types and then assign these groups to balance the workload. Ammons, Lofgren, and McGinnis (1985) address the problem of allocating components to workstations to balance workloads and to minimize workstation visits in an environment where an assembly visits a workstation only once. Workload balancing has also been studied by Fathi and Taheri (1989) and Rajan and Segal (1989) with a secondary objective of minimizing the number of machines to be visited. Crama, Kolen, Oerlemans, and Spieksma (1990) describe a hierarchical decomposition and mathematical programming approach for a line with several placement machines which are devoted to the assembly of a single product with restrictions on the component types that can be loaded.

In a recent paper, Balakrishnan and Vanderbeck (1999) consider a partial setup strategy of mounting frequently used components permanently on each machine and present a model to assign product families to parallel assembly lines incorporating both workload balancing and setup time minimization objectives. Partial setup strategies have been discussed by Leon and Peters (1996) in the case of a single placement machine to minimize the makespan and by Peters and Subramanian (1996) in the case of multiple placement machines operating in parallel to balance the tradeoff between processing time and changeover time.

Ahmadi, Grotzinger, and Johnson (1988) and Tang (1988) propose the use of duplicate reels of each component type. Additional feeder tapes of the same component can be placed onto different machines to balance the workload or onto the same machine to improve the cycle time. Although this practice may improve the solution in certain cases, it requires additional investment in parts, planning, and setup.

Lofgren, McGinnis, and Tovey (1991) discuss the component allocation problem with precedence constraints and prove that the general problem of minimizing the number of workstation visits is NP-hard. They also show that standard heuristics have arbitrarily bad worst case performance. Askin, Dror, and Vakharia (1994) consider the case where precedence constraints do not exist. They develop a methodology to group PCBs into production families, allocate component types to machines, divide families into board groups with similar processing times, and schedule the groups in order to minimize the makespan and reduce the mean flow time. In an environment where precedence constraints on the order of component placements do not exist, the machine scheduling problem can be studied in the context of an openshop scheduling problem. Since our problem setting is similar, we overview the openshop scheduling literature next.

The openshop machine scheduling problem is defined as follows: n jobs (J_1, ..., J_n) have to be processed by m machines (M_1, ... ,M_m). Job J_i, i=1, ..., n, requires at most m operations (O_{i1}, ... , O_{im}), where O_{ir} is the operation performed on machine r for job i. Operation O_{ir} has to be processed (uninterrupted if preemption is not allowed) for t_{ir} time units on machine M_r, r=1, ... , m. t_{ir} is a nonnegative integer. In an openshop,

there is no restriction on the order in which the operations of a job are to be performed. Each job can be processed on at most one machine at a time and each machine can process at most one operation at a time.

In this section, the three-fold notation $\alpha \mid \beta \mid \gamma$ of Lawler, Lenstra, Rinnooy Kan, and Shmoys (1989) is used. The first field α specifies the machine environment, the second field β indicates the job characteristics, and the third field γ refers to the optimality criterion. In our case, the problem is indicated as $Om \mid t_{ir} = 1 \mid C_{max}$ where O stands for openshop discipline and m for the number of machines. If the number of machines is arbitrary then m is omitted. C_i denotes the completion time of job J_i and $C_{max} = max_i\ C_i$ is the completion time of the last job in the schedule.

Dror (1992) examines openshop scheduling problem with machine dependent processing times and provides a polynomial time algorithm for $O \mid t_{ir} = t_r,\ n \geq m \mid C_{max}$ and $O \mid t_{ir} = t_r,\ t_1 \geq 2t_2 \geq \ldots \geq 2t_m,\ \sum_{r=2}^{m} t_r \leq t_1 \mid \sum C_i$. He also proves that $O \mid t_{ir} = t_r,\ m \geq 3,\ n < m \mid C_{max}$ is NP-hard and gives a linear time algorithm for $O2 \mid t_{ir} = t_r \mid \sum C_i$. Vakharia and Çatay (1997) show that Dror's (1992) algorithm is not optimal and propose a refinement of the procedure while proving its optimality. Askin, Dror, and Vakharia (1994) address the PCB family formation and scheduling problem in an openshop environment with a primary objective of minimizing the makespan for assembling a batch of boards and a secondary objective of reducing the mean flow time.

In our study, t_{ir}'s (i.e., insertion time for PCB i on machine M_r) are estimated by the component requirements of the PCBs based on the assignments of the component

families on machine M_r. Hence, the problem is that of assigning component types to the machines (to estimate t_{ir}) and scheduling the PCBs. This scheduling environment can be characterized in the context of a flexible openshop since all insertion machines are identical and have the flexibility of processing any PCB type. In that sense, the machines become distinct only after the component families are assigned. Thus, family formation plays a critical role in this setting where the grouping of component types dictates the type of schedule generated. Further, the completion time of jobs is determined by the assignments of families to machines since the processing time is a function of the number

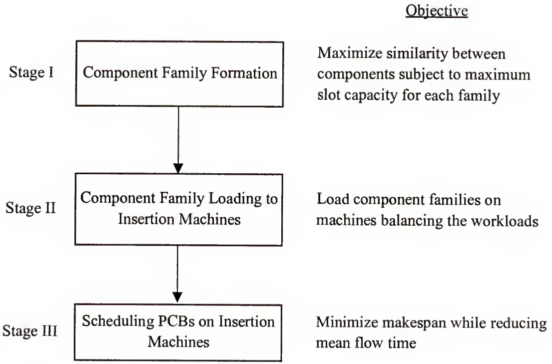


Figure 4.1. Overview of the proposed method

of components to be loaded on each board. In order to achieve a short makespan, workload balancing between machines needs also to be considered when assigning the component families. A lower bound on the makespan is given by the maximum amount of components to be placed by any machine and this amount is minimized in the case of perfectly balanced machines.

4.3. An Integrated Method for Component Family Formation, Component Loading, and PCB Scheduling

In this section, we present a three stage method to group components into families, to load component families on insertion machines, and to schedule PCBs. An overview of this method is shown in Figure 4.1. In the first stage, component families are created in order to maximize the similarity between components without exceeding maximum slot capacity for each family. In the second stage, we load component families on the insertion machines attempting to balance the workloads. In the third stage, we schedule PCBs to machines in order to minimize the makespan with the secondary objective of reducing the mean flow time. We now proceed to describe details of each stage.

4.3.1. Stage I: Component Family Formation

A comprehensive 0-1 integer programming formulation of a component family formation problem is as follows:

$$\max \quad \sum_{i \in R} \sum_{j \in P} S_{ij} x_{ij} \tag{1}$$

$$\text{s.t.} \quad \sum_{j \in P} y_j = p, \tag{2}$$

$$\sum_{j \in P} x_{ij} = 1, \qquad i \in R \tag{3}$$

$$\sum_{i \in R} f_i \, x_{ij} \le C, \qquad j \in P \tag{4}$$

$$x_{ij} \le y_j, \qquad i \in R, j \in P \tag{5}$$

$$x_{ij}, y_j \in \{0,1\}, \qquad i \in R, j \in P \tag{6}$$

where i,j are indices for component types, R and P are the set of component types and medians, respectively. S_{ij} is the similarity coefficient, p is the number of medians (number of families) to be identified, f_i is the number of feeder slots required by component type i, and C is the capacity of each machine. The decision variables are

$$x_{ij} = \begin{cases} 1, & \text{if component } i \text{ is assigned to family j} \in \text{P} \\ 0, & \text{otherwise} \end{cases}$$

$$y_j = \begin{cases} 1, & \text{if component } j \text{ is the median for family j} \in \text{P} \\ 0, & \text{otherwise} \end{cases}$$

The objective function (1) maximizes the sum of similarities between each pair of component types. Constraint (2) sets the number of component families; constraint set (3) ensures that all component types are assigned to a single family; constraint set (4) enforces the feeder slot capacity on the machines and constraints (5) ensure assignment

of a component to a family that is created. Thus, the above clustering problem's objective is to construct p families that are as homogeneous as possible, while not violating the capacity requirements of the machines.

Since the optimal 0-1 integer programming formulation described above is not practically tractable, we propose an efficient grouping algorithm to derive good solutions to this problem. An overview of the three major steps in our procedure is as follows:

- First, a set of p medians is obtained using an initial heuristic. Two different heuristics are used for the initialization process.
- Second, component types are assigned to their most similar median without violating the feeder slot capacity. After all components are assigned, the similarities between each component and all other components assigned to the same median are computed. The set of medians is updated if a higher similarity level is achieved by a component other than the median.
- Finally, after the above process is complete, all possible interchanges between pairs of component types assigned to two different families are considered. If an interchange can improve the objective function value without violating the capacity constraint, the two component types are swapped. The procedure is repeated until no further feasible exchange is possible.

Before describing each step of the procedure, we first define the following additional notation:

F	number of component families (may or may not be prespecified)
NF	number of component families at each iteration
p_{ik}	k^{th} most similar median to component type i, k=1,2,3,...
$CurrLoad(k)$	current feeder slots required by all components types assigned to median k
FS_i	feeder slots required by component type i
OBJ	objective function value
P_{Temp}	temporary set of medians
$SMS(P)$	sum of the maximum similarities between all pairs of $i \in R$ and $j \in P$

$$= \begin{cases} \sum_{\substack{i \in R \\ i \neq j}} \max_{j \in P} S_{ij} & \text{for } P \subset R \\ -\infty & \text{for } P = \varnothing \end{cases}$$

SS_{ik}	sum of similarities between component i and all components where i is assigned to median k

$$= \sum_{\substack{j \in R \\ j \neq i}} S_{ij}\, x_{ik}\, x_{jk}\ , \qquad i \in R, k \in P, i \neq k$$

4.3.1.1. Initialization Algorithms

Two heuristics are used to form the initial set of medians. The first is a greedy algorithm where, for a given set of medians, the component type that gives the greatest immediate increase in the value of the sum of the similarities is chosen as the next

component to be included in the median set. In the second algorithm, the pair of component types that has the greatest similarity are merged into a new component type, provided that the number of feeder slots required by both does not exceed the machine's capacity. If the total number of medians is prespecified, the algorithms will stop when this number is reached. Otherwise they will stop when the value of the sum of similarities fails to increase.

Initial Algorithm 1 (IA I):

Step 0: Let $P^0 = \varnothing$, $NF = 0$, $t = 1$.

Step 1: Let $j_t = \arg\max_{j \in R \setminus P^{t-1}} SMS(P^{t-1} \cup \{j\})$

Step 2: If $SMS(P^{t-1} \cup \{j\}) \leq SMS(P^{t-1})$, stop. P^{t-1} is an initial median set.

Otherwise, set $P^t = P^{t-1} \cup \{j_t\}$ and let $NF = NF + 1$.

Step 3: If $P^t = R$, or if $NF = F$ in case F is prespecified, stop. R is an initial median set. Otherwise, let $t = t + 1$, go to Step 1.

Initial Algorithm 2 (IA II):

Step 0: Let $P^0 = \varnothing$, $NF = |R|$, $t = 1$.

Step 1: Identify i_t and j_t such that $S_{ij} = \max_{i<j} \{S_{ij}\}$

Step 2: If $S_{ij} = 0$, stop. P^{t-1} is an initial median set.

Otherwise, compute total number of feeders required by i_t and j_t: TF

Step 3: If $TF > C$ let $S_{ij} = 0$, go to Step 5.

Otherwise, go to Step 4.

Step 4: If $NF \leq F$, stop. P^{t-1} is an initial median set.

Otherwise go to Step 5.

Step 5: Set $NF = NF$-1; merge i_t and j_t to create a new temporary component type k_t. Let $P^t = P^{t-1} \cup \{j_t\}$. Let $t = t$ +1, go to Step 1.

4.3.1.2. Grouping Algorithm

After initial set of medians is determined component types are assigned to their most similar median as long as the machine slot capacity is not exceeded. The assignment procedure is as follows: for each component type, the absolute values of the differences between the similarity coefficients of the component and its first and second most similar medians are computed. Then, component types are assigned to medians in non-increasing order of these absolute values. By doing this, the assignment of a component to a very dissimilar median is prevented.

When all of the assignments are completed for each component type in every family the sum of similarities between that component and other components assigned to the same median is computed. If a component type other than the median achieves the maximum sum of similarities then the set of medians is updated by adding that component type which improves the objective function value and removing the old median. The method is repeated with the new set of p medians as long as the objective function value continues to improve with a new component type.

The details of the procedure to group components into families are as follows:

Step 0: Let $P^1 = P^t$ from the initial algorithm and set $t = 1$.

Step 1: For each component type i, sort median j's in non-increasing order of S_{ij}, $i \in R, j \in P$.

Step 2: Compute $DIF_i = | S_{i,p_{i1}} - S_{i,p_{i2}} |$. Sort DIF_i's in non-increasing order. Set $k = 1$, go to next step.

Step 3: If component type i is not assigned and $CurrLoad(p_{ik}) + FS_i \leq C$, assign i to median p_{ik}. Let $CurrLoad(p_{ik}) = CurrLoad(p_{ik}) + FS_i$, go to Step 4. Otherwise, let $k = k + 1$, go to Step 3.

Step 4: Set $P_{Temp} = \varnothing$. Compute SS_{ik}'s.

Step 5: Find the greatest SS_{ik}. If i is not a median, replace median k with component type i: i becomes the new median. Set $P_{Temp} = P_{Temp} \cup \{i\}$. Go to Step 6.

Step 6: Compute OBJ. If $P_{Temp} = P^t$, stop. Otherwise, let $P^t = P_{Temp}$, $t = t + 1$, go to Step 1.

4.3.1.3. Improvement Algorithm

We consider all possible component exchanges between two families to further improve the similarity between components. The algorithm is as follows:

Step 0: Let $i,j \in R$ and $k,l \in P$

Step 1: If $(S_{ik} + S_{jl}) > (S_{il} + S_{jk})$ and $CurrLoad(k) + FS_i - FS_j \leq C$ and $CurrLoad(l) + FS_j - FS_i \leq C$, switch component types i and j between median k and median l.

Step 2: Update $CurrLoad(k)$ and $CurrLoad(l)$.

If all components and medians are considered, stop.

Otherwise, go to Step 0.

4.3.2. Stage II: Component Family Loading

Component families are allocated to machines attempting to keep the workload on the most utilized machine as low as possible. First, we compute the processing time needed to place all the components in each family onto the PCBs that require them. Then component families are loaded on machines such that total number of feeders required does not exceed slot capacity C and maximum of the total processing times is minimized to achieve a balanced workload. The algorithm is as follows:

Step 0: Compute the total processing time of each component family.

Step 1: Load families for which total feeder slot requirements are at capacity C. (dedicating the machine to a unique component family)

Step 2: Set the upper bound on the makespan to the largest processing time among all families.

Step 3: Load sequentially the remaining component families to the next available machine up to the upper bound on the makespan. Skip families for which the machine does not have sufficient feeder slot capacity.

Step 4: If the machine is loaded up to its slot capacity, repeat Step 3 for the remaining machines.

Step 5: If there exist unloaded component families, increase the upper bound on the makespan by the smallest processing time among all remaining families. Go to Step 3.

4.3.3. Stage III: Scheduling PCBs on Machines:

Given that the primary objective is to minimize the maximum completion time with a secondary objective of reducing the mean flow time, PCB scheduling is carried out as follows. First, we assign PCBs to machines based on their component requirements. After all of the assignments are completed, we create a PCB-Machine Processing Time matrix where each entry represents the time based on the number of components each PCB requires. Next, the machines are sorted in the non-increasing order of their total workload: $\sum_{i=1}^{n} t_{i1} \geq \sum_{i=1}^{n} t_{i2} \geq \ldots \geq \sum_{i=1}^{n} t_{im}$. The logic behind this Longest Processing Time (LPT) type ordering is that the machine currently having maximum workload is the bottleneck in the system and boards are scheduled to that machine first so that it will have no idle time (in the case of the first machine) or the least amount of idle time (in the case of the following machines). This LPT type ordering is employed in an

effort to minimize the makespan. Then, the boards are sorted in the non-decreasing order of their machine processing times. This Shortest Processing Time (SPT) type ordering is applied to reduce the mean flow time and PCBs are then scheduled on each machine. If a PCB is currently assigned to a previous machine, it is scheduled at an earlier time if the machine has sufficient idle time. Otherwise, it is delayed until its previous operation has been completed.

The scheduling algorithm is performed as follows:

Step 0: Assign PCBs to machines where the components they require have been loaded and create PCB-Machine Processing Time matrix.

Step 1: Sort the machines in non-increasing order of the total processing times. (LPT rule for machines)

Step 2: Sort PCBs on each machine in non-decreasing order of their processing time on that machine. (SPT rule for PCBs)

Step 3: Starting with the machine that has longest processing time, i.e. starting with the first machine in the sequence, schedule PCBs in the SPT order. If a PCB is being processed on a previous machine, schedule it to an earlier time if the machine has sufficient idle time. Otherwise, delay it until its previous operation has been completed.

4.4. Experimental Analysis

4.4.1. Industrial Setting

The methodology described in the previous section to form component families and to schedule PCBs to the insertion machines was coded in Pascal programming language. The data were obtained from a major electronics company and consist of information regarding 22 different PCB types they manufacture and 290 distinct component types. The data include component types required by each PCB type along with the number of units of component types to be mounted on the PCB and the number of feeder slots required on the insertion machines to hold the component type. The number of slots is 3, 6, or 9. There are eight automatic surface mount placement machines that are identical and the total number of feeder slots available on each machine is 180.

We first reduce the component set by combining components required by the same set of PCBs. This helps to reduce the memory requirements and run time, and to prevent unnecessary data handling. Components which are to be placed on the same PCB or PCBs are merged into a new component type, provided that the sum of the number of slots required by the new component does not exceed the machine capacity of 180 slots that are available. By doing so, the component set is decreased to 74 distinct component types and slot requirements of the new component types and the number of units mounted on each PCB are updated. Table 4.1 shows the information for PCB type #22 that originally required 11 distinct component types and now requires 10 after the reduction procedure.

For each i and j, $i, j \in R$, and $i \neq j$, similarity coefficients (S_{ij}) could be defined in a variety of ways. In our case, we defined them as follows:

- $$S_{ij}^{1} = \frac{NCB_{ij}}{TNB_i + TNB_j - NCB_{ij}}$$

NCB_{ij} is the number of PCBs requiring both component types i and j, TNB_i is the number of PCBs requiring component type i and similarly TNB_j is the number of PCBs requiring component type j.

- $$S_{ij}^{2} = \sum_{b \in B} (UM_{ib} - UM_{jb})^2$$

UM_{ib} is the number of units of component i mounted on PCB b, UM_{jb} is the number of units of component j mounted on PCB b, and B is the set of PCBs requiring either component i or j or both.[1]

- $$S_{ij}^{3} = \frac{\sum_{b \in B} \frac{min\{UM_{ib}, UM_{jb}\}}{max\{UM_{ib}, UM_{jb}\}}}{NCB_{ij}}$$

The primary reason for defining similarity coefficients in several ways was to investigate whether the families identified are significantly impacted by this parameter

---

[1] S_{ij}^{2} is in fact a dissimilarity coefficient. In that case, the problem becomes a minimization problem.

Table 4.1. Component information for PCB type # 22

Component No.	Units Mounted	Feeder Slots Required
1	1	6
3	2	6
5	1	6
13	4	9
15	3	6
27	46	6
46	20	6
60	7	6
71	1	9
74	74	12

choice. In what follows a numerical example, illustrating the computation of the three similarity coefficients discussed earlier, is given. Table 4.2 lists PCBs that require component 46 and component 60 along with the number of component units to be mounted. Noting that the number of PCBs that require component type 46 is 8, the

Table 4.2. Data on PCB types requiring components #46 and #60

Component No. 46	
PCB Type	Number of units to be mounted
8	8
10	10
11	3
14	2
16	2
20	14
21	20
22	20

Component No. 60	
PCB Type	Number of units to be mounted
11	5
14	8
15	18
16	8
18	1
19	1
20	2
21	7
22	7

number of PCBs that require component type 60 is 9 and the number of PCBs that require both component types is 6, the similarity coefficients between these two component types are calculated as follows:

$$S^{1}_{46,60} = \frac{6}{8+9-6} = 0.545,$$

$$S^{2}_{46,60} = \frac{\frac{0}{8}+\frac{0}{10}+\frac{3}{5}+\frac{2}{8}+\frac{0}{18}+\frac{2}{8}+\frac{0}{1}+\frac{2}{14}+\frac{7}{20}+\frac{7}{20}}{6} = 0.324,$$

$$S^{3}_{46,60} = (8-0)^2+(10-0)^2+(3-5)^2+(2-8)^2+(0-18)^2+(2-8)^2+$$
$$(0-1)^2+(0-1)^2+(14-2)^2+(20-7)^2+(20-7)^2 = 1048$$

4.4.2. Grouping Component Types and Scheduling PCBs

In order to identify the median components for each component family, the two initialization algorithms (IA I and IA II) are used. The algorithms require the user to prespecify the number of medians, hence the number of families to be formed. In the first algorithm, if the user does not to prespecify number of component families the algorithm stops identifying new medians when the sum of similarities between the component pairs in each family reaches to maximum. At the end, both algorithms identify a number of components that will be used as the initial median set in the grouping algorithm. Random generation of components as initial medians was also considered. However, heuristics IA I and IA II provided better performance both in terms of quality of the results measured by the similarity criterion and execution time.

After the initial medians are identified, grouping and improvement algorithms determine component families for the specified number of medians as described in Stage I. A large range of families (8 to 40) has been examined for experimental purposes. The reason for doing so is that the component family population and number of families may have a significant affect on the assignment of components to machines, therefore on the schedule. Since the processing times for PCB types on each machine differ significantly and component family formation algorithms do not require any intensive computations it may be useful to investigate different number of families to compare the performance results.

After all component families are identified, we compute total number of component units to be mounted, i.e., total processing time, for each family. Next, all families requiring 180 feeder slots are loaded to insertion machines. Then, we find the maximum processing time among all component families as an upper bound on the workload on each machine and start loading remaining families in the order of their formation up to that upper bound. The same procedure is repeated gradually increasing the upper bound by the smallest processing time among all remaining families. Regarding the capacity of machines note that total number of feeder slots required by all components is 1208 compared to a total capacity of 1440 (8 machines times 180) slots.

Finally, when all component families are loaded, a lower bound on the makespan is obtained as the total processing time on the machine mounting the maximum number of component units among all machines. PCBs are then assigned to machines based on

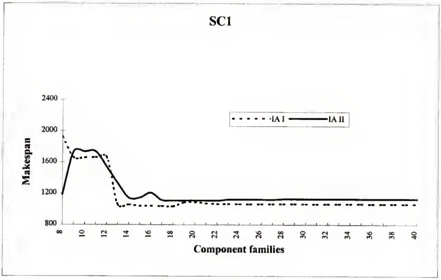
their component requirements and scheduled following the algorithm in Stage III described in the previous section.

4.4.3. Computational Results

The charts in Figure 4.2, 4.3, and 4.4 show the completion times and mean flow times obtained for different number of component families using three different similarity coefficients (SC). Shorter completion times are achieved when the number of component families varies between 14 and 20, except in the case of SC1 where shortest completion time is achieved forming 36 component families using IA II; when IA I is used makespan stays the same for number of families greater than 12. The utilization of SC2 and SC3 results in shorter completion times for almost all numbers of families compared to SC1. Furthermore, SC2 outperforms SC3 in terms of both makespan and mean flow time. Regarding the performance of the Scheduling Algorithm, out of 33 different numbers of

Table 4.3. Summary of results for IA I

	IA I				
	Number of Families	Makespan	Gap	MFT	Number of Visits
SC 1	18	1033	15.97%	499	77
SC 2	14	893	2.80%	443	96
SC 3	13	992	12.50%	465	103



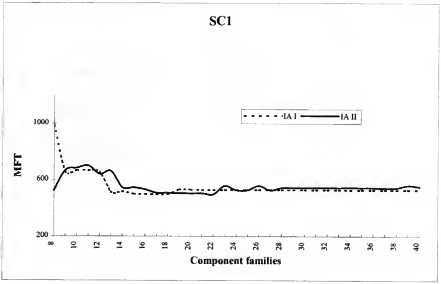
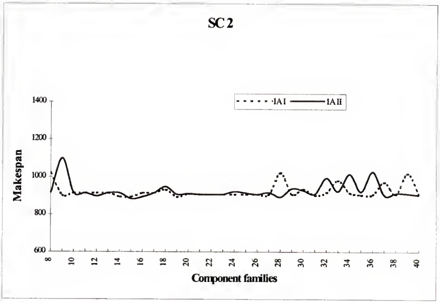


Figure 4.2. Makespan and mean flow time using similarity coefficient 1



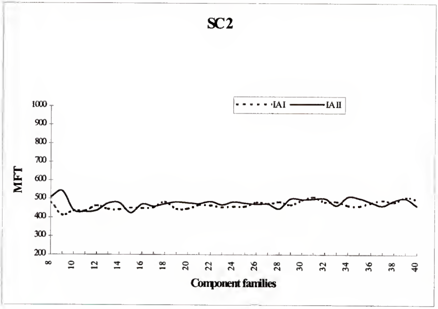
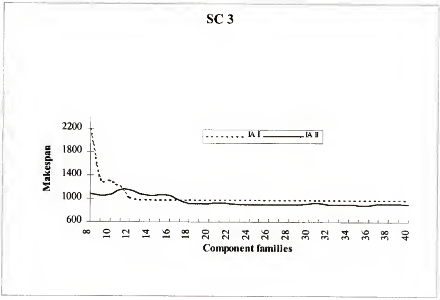


Figure 4.3. Makespan and mean flow time using similarity coefficient 2



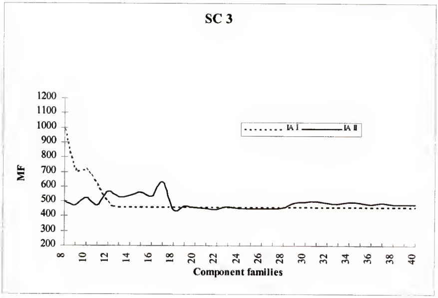


Figure 4.4. Makespan and mean flow time using similarity coefficient 3

families examined, the lower bound on the makespan is achieved in 43 instances using SC1 (10 with IA I and 33 with IA II), 57 using SC2 (28 with IA I and 29 with IA II), and 64 using SC2 (32 with IA I and 32 with IA II). The deviation from the lower bound is 1.14% on the average and 12.85% in the worst case. The results obtained using two initialization algorithms are summarized in Table 4.3 and Table 4.4. All of the completion times are at their lower bounds for the component families created. The percent difference between the makespan found and the minimum makespan is indicated in the Gap column. Optimal makespan (of 868) is found by solving the following Min-Max Generalized Assignment Problem:

$$
\begin{aligned}
\min \quad & z \\
\text{s.t.} \quad & \sum_{i \in P} t_i x_{ij} \le z, && j \in M \\
& \sum_{j \in M} x_{ij} = 1\,, && i \in P \\
& \sum_{i \in P} f_i \, x_{ij} \le C, && j \in M \\
& x_{ij} \in \{0,1\}, && i \in P, j \in M
\end{aligned}
$$

where M is the set of machines

Component family formation reduces the number of machines to be visited substantially. In all of the instances, fewer number of visits is achieved compared to 115 total machine visits (5.23 machines/PCB on the average) in the optimal solution. In the

Table 4.4. Summary of results for IA II

	IA II				
	Number of Families	Makespan	Gap	MFT	Number of Visits
SC 1	22	1106	21.52%	498[a]	84
SC 2	15	883	1.70%	425[a]	94
SC 3	36	894	2.91%	480	83

[a] indicates that the a smallest MFT is obtained among all component family numbers examined.

case where the shortest makespan is obtained, the average number of visits is 4.27 whereas the average is 3.77 in the best case.

In sum, the results indicate that component grouping using SC2 with either IA I or IA II provides near-optimal solutions and better number of machine visits for the environment studied. The grouping and scheduling algorithms developed may be applied in other openshop environments where component mounting times do not depend on the location of the component feeders on the machine, no setup is needed when switching from one PCB to another, and machines are identical in terms of placement times and number of feeder slots required for each component.

CHAPTER 5
CONCLUSIONS AND DIRECTIONS FOR FUTURE RESEARCH

In this research, we first presented a mathematical model for assigning operations to machine groups and determining additional machine requirements. We have proposed a Lagrangian relaxation/decomposition based solution procedure given the intractability of the model. Our approach is shown to provide near optimal solutions in short CPU times. Our methodology may be applied/adapted for other similar manufacturing applications where resource allocation and capacity expansion decisions are to be made.

This first model is extended to consider multi-period production planning by including inventory variables and inventory balance constraints between consecutive periods to guarantee consistency of production volumes. In that context, we propose a mathematical model for alternative machine type capacity allocation problem with machine duplication in semiconductor manufacturing and a Lagrangian-based heuristic solution procedure to solve this problem. Our approach is shown to provide fairly good feasible solutions in a relatively short amount of time without necessitating the use of any optimization tools. The solution may as well be used as an initial solution to obtain an optimal or better solution employing an optimization software.

The method may be applied to production and capacity planning problems of other industries where resource allocation and new equipment acquisition decisions are to be made. Demand for new equipment arises primarily from two sources: replacement of

existing aged equipment, and additional equipment required to meet growth in demand for the firm's products and services. The model discussed here can also be extended to consider all processes in the manufacture of a computer chip by including constraints between consecutive process steps to guarantee consistency of production volumes. Furthermore, backorders may also be considered in that context. The approach could also be extended to investigate replacement of capacity as well as expansion and disposal to adapt to arbitrary changes in demand.

A limitation of the model is that it assumes deterministic technological changes, i.e., all technologies available in the future are assumed to be known at the beginning. Other limitations include assumptions of space availability in the facilities for additional equipment and accurate forecasts of future costs and demands.

Finally, we present an integrated methodology for grouping components into families, loading these families of components on individual insertion machines, and finally, scheduling PCBs on the machines. Our integrated approach is shown to provide near-optimal solutions in a reasonable period of time. Further, when comparing it to a procedure where we simply load individual components to machines rather than loading component families on machines, we show that the creation of families results in fewer visits of PCBs to individual machines (leading to a reduction in materials handling costs).

We also show that using our approach, the user can determine the "best" number of families that should be formed based on scheduling measures rather than the structural measures used in prior work on cellular manufacturing. This, in our opinion, is a much

more effective approach for determining the number of families since it looks at how actual shop performance measures.

Extensions of this study could focus on several aspects. First, more effective algorithms could possibly be developed for solving the component grouping problem addressed in Stage 1 of our method. Second, although we have only experimented with industry data in PCB manufacturing, the approach could also be extended/adapted for other similar manufacturing applications such as machine tool manufacturing which is also characterized by high component variety and few final products.

APPENDIX A
EXPERIMENTAL STUDY RESULTS FOR OPERATION/MACHINE TYPE ASSIGNMENT METHODOLOGY

The results of the experimental analysis discussed in Chapter 2 are illustrated here. The numbers on the upper left corner of each table indicate number of machines and number of operations, respectively. Z* is the optimal value achieved at each run, LB is the lower bound on the minimum value of the objective function, and UB is the upper bound on the minimum value of the objective function. Thus, the percent value is the gap between the optimum value and the lower bound, and the upper bound and the optimum value, respectively. The average is also shown at the bottom of each set of five runs. The final table displays overall means of six problem types. This table excludes problem type with 40 machine groups and 400 operations since the optimal solutions are not available.

(4,92)		Variable Cost	Machine Cost							
			(10, 12)		(10, 15)		(10, 20)		(10, 40)	
			(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*
Utilization	0.60	(0.1, 0.3)	0.003%	4.003%	0.346%	5.338%	0.479%	3.217%	1.978%	0.000%
			0.008%	0.052%	0.016%	0.110%	1.575%	0.035%	1.188%	0.000%
			0.006%	0.130%	0.305%	0.000%	0.031%	0.165%	2.103%	0.000%
			0.004%	0.000%	0.050%	0.000%	0.602%	0.019%	1.528%	2.076%
			0.014%	0.048%	0.026%	3.784%	0.113%	0.000%	1.479%	0.000%
			0.007%	0.847%	0.149%	1.846%	0.560%	0.687%	1.655%	0.415%
		(0.1, 0.9)	0.011%	2.989%	0.089%	0.007%	0.936%	1.664%	0.019%	0.000%
			0.016%	0.000%	0.006%	0.005%	0.022%	0.449%	0.068%	0.000%
			0.058%	0.000%	0.058%	0.070%	0.041%	0.000%	0.016%	0.363%
			0.022%	0.138%	0.047%	0.000%	0.002%	0.000%	0.052%	0.000%
			0.112%	3.023%	0.003%	0.058%	0.043%	0.000%	0.116%	0.017%
			0.044%	1.230%	0.041%	0.028%	0.209%	0.423%	0.054%	0.076%
		(0.1, 1.9)	0.018%	0.000%	0.035%	0.032%	0.098%	0.000%	0.077%	0.916%
			0.023%	0.106%	0.016%	0.000%	0.058%	0.000%	0.034%	0.078%
			0.293%	1.219%	0.061%	0.000%	0.250%	0.112%	0.011%	0.116%
			0.015%	2.259%	0.012%	0.161%	0.122%	0.066%	0.084%	4.311%
			0.055%	1.734%	0.085%	0.000%	0.048%	0.164%	0.002%	0.117%
			0.081%	1.064%	0.042%	0.039%	0.115%	0.068%	0.042%	1.108%
	0.75	(0.1, 0.3)	0.000%	0.086%	0.467%	0.125%	0.007%	0.042%	0.135%	0.000%
			0.016%	0.000%	1.293%	0.108%	0.009%	0.085%	1.318%	0.000%
			0.032%	0.161%	0.020%	0.039%	0.003%	0.147%	0.542%	0.000%
			0.015%	0.024%	0.005%	0.000%	0.025%	0.252%	0.020%	0.705%
			0.012%	0.000%	0.176%	0.091%	0.010%	0.000%	0.155%	0.000%
			0.015%	0.054%	0.392%	0.073%	0.011%	0.105%	0.434%	0.141%
		(0.1, 0.9)	0.085%	4.320%	0.002%	0.331%	0.025%	0.382%	0.012%	0.435%
			0.054%	0.006%	0.010%	0.081%	0.003%	3.905%	4.447%	0.087%
			0.007%	0.000%	0.059%	0.000%	1.108%	0.000%	0.037%	0.752%
			0.056%	0.000%	0.056%	0.179%	0.107%	0.000%	0.042%	0.952%
			0.015%	0.000%	0.057%	0.121%	0.028%	0.000%	0.333%	0.000%
			0.043%	0.865%	0.037%	0.142%	0.254%	0.857%	0.974%	0.445%
		(0.1, 1.9)	0.057%	0.060%	0.245%	5.529%	0.174%	0.031%	0.058%	0.135%
			0.082%	0.000%	0.046%	0.198%	0.088%	0.049%	0.167%	0.000%
			0.039%	0.000%	0.059%	0.000%	0.016%	0.183%	0.682%	0.116%
			0.030%	0.000%	0.051%	3.554%	0.050%	0.691%	0.171%	0.000%
			0.018%	0.000%	0.073%	0.000%	0.064%	0.210%	3.894%	0.000%
			0.045%	0.012%	0.095%	1.856%	0.079%	0.233%	0.994%	0.050%
	0.90	(0.1, 0.3)	0.042%	0.113%	0.002%	0.036%	0.000%	0.000%	0.000%	0.000%
			0.012%	7.437%	0.033%	0.000%	0.000%	0.000%	0.000%	0.000%
			0.001%	0.121%	0.028%	0.009%	0.001%	0.057%	0.032%	0.000%
			0.018%	0.208%	0.005%	0.000%	0.000%	0.000%	0.000%	0.000%
			0.021%	0.000%	0.017%	0.114%	0.879%	0.000%	0.027%	0.000%
			0.019%	1.576%	0.017%	0.032%	0.176%	0.011%	0.012%	0.000%
		(0.1, 0.9)	0.102%	0.020%	0.017%	0.000%	0.160%	0.100%	0.043%	0.048%
			0.076%	5.781%	0.059%	0.000%	0.008%	0.462%	0.019%	0.000%
			0.011%	0.000%	0.034%	0.000%	0.108%	0.000%	0.024%	0.000%
			0.010%	0.174%	0.020%	0.296%	0.090%	0.402%	0.032%	0.000%
			0.015%	0.000%	0.095%	0.210%	0.042%	0.000%	0.000%	0.000%
			0.043%	1.195%	0.045%	0.101%	0.082%	0.193%	0.023%	0.010%
		(0.1, 1.9)	0.064%	0.000%	0.067%	0.000%	0.017%	0.000%	0.042%	0.796%
			0.008%	0.000%	0.209%	3.888%	0.051%	0.035%	0.239%	3.548%
			0.098%	4.132%	0.003%	0.043%	0.136%	0.085%	0.182%	0.000%
			0.060%	0.152%	0.050%	4.028%	0.164%	0.016%	0.052%	0.000%
			0.091%	0.039%	0.135%	0.000%	0.009%	0.472%	0.049%	0.610%
			0.064%	0.865%	0.093%	1.592%	0.076%	0.122%	0.113%	0.991%

(4,120)		Variable Cost	Machine Cost							
			(10, 12)		(10, 15)		(10, 20)		(10, 40)	
			(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*
Utilization	0.60	(0.1, 0.3)	0.049%	4.479%	0.037%	5.024%	0.394%	7.020%	0.501%	6.214%
			0.004%	0.179%	0.052%	0.000%	0.015%	0.120%	0.242%	2.420%
			0.108%	0.102%	0.026%	3.911%	0.846%	4.224%	1.502%	7.582%
			0.003%	0.081%	0.072%	0.000%	0.184%	6.234%	2.448%	1.904%
			0.047%	0.051%	0.023%	0.359%	0.297%	1.601%	1.484%	1.170%
			0.042%	0.978%	0.042%	1.859%	0.347%	3.840%	1.235%	3.858%
		(0.1, 0.9)	0.016%	0.000%	0.016%	0.000%	0.014%	0.000%	0.759%	0.492%
			0.112%	0.050%	0.135%	1.903%	0.047%	3.973%	0.246%	2.008%
			0.066%	0.294%	0.102%	1.507%	0.119%	2.053%	0.173%	1.895%
			0.035%	0.172%	0.098%	0.000%	0.221%	0.000%	1.009%	0.005%
			0.025%	5.818%	0.020%	4.776%	0.032%	0.754%	2.752%	0.728%
			0.051%	1.267%	0.074%	1.637%	0.086%	1.356%	0.988%	1.026%
		(0.1, 1.9)	0.092%	1.914%	0.057%	0.216%	0.042%	0.983%	0.275%	0.000%
			0.279%	1.024%	0.225%	0.519%	0.428%	1.178%	0.390%	5.809%
			0.242%	0.000%	0.052%	1.434%	0.487%	0.965%	0.144%	0.046%
			0.685%	0.659%	0.059%	2.166%	0.253%	3.013%	0.565%	0.000%
			0.138%	2.342%	0.055%	1.076%	0.493%	0.070%	0.163%	0.000%
			0.287%	1.188%	0.090%	1.082%	0.341%	1.242%	0.308%	1.171%
	0.75	(0.1, 0.3)	0.101%	0.521%	0.192%	1.042%	1.313%	0.800%	2.740%	3.502%
			0.041%	0.114%	0.287%	6.564%	0.507%	2.432%	0.543%	1.396%
			0.046%	0.375%	0.134%	2.022%	1.419%	0.000%	0.245%	2.402%
			0.046%	0.000%	0.009%	0.056%	0.506%	0.000%	1.538%	0.000%
			0.041%	0.044%	0.006%	0.212%	0.591%	0.000%	0.015%	0.000%
			0.055%	0.211%	0.126%	1.979%	0.867%	0.646%	1.016%	1.460%
		(0.1, 0.9)	0.038%	0.319%	0.058%	0.306%	0.415%	0.308%	1.045%	3.599%
			0.023%	0.359%	0.568%	2.773%	0.192%	0.086%	1.509%	0.000%
			0.136%	0.000%	0.154%	0.931%	0.288%	3.228%	2.630%	2.449%
			0.112%	0.192%	0.123%	3.842%	0.324%	0.951%	0.788%	5.895%
			0.111%	0.310%	0.021%	0.139%	0.031%	0.444%	0.051%	1.836%
			0.084%	0.236%	0.185%	1.598%	0.250%	1.003%	1.204%	2.756%
		(0.1, 1.9)	0.062%	0.000%	0.001%	0.000%	0.186%	2.252%	0.990%	2.282%
			0.037%	0.371%	0.012%	0.215%	0.215%	3.088%	0.080%	0.716%
			0.020%	3.158%	0.056%	0.079%	0.089%	0.370%	0.041%	0.000%
			0.043%	1.380%	0.188%	0.451%	0.051%	0.000%	1.612%	0.063%
			0.196%	2.592%	0.023%	0.000%	0.110%	0.142%	1.834%	4.428%
			0.071%	1.500%	0.056%	0.149%	0.130%	1.170%	0.911%	1.498%
	0.90	(0.1, 0.3)	0.100%	6.454%	2.872%	6.358%	3.760%	3.810%	2.964%	0.000%
			0.053%	6.976%	0.539%	4.922%	3.245%	8.811%	0.522%	7.021%
			0.400%	0.000%	0.064%	0.000%	0.427%	6.654%	2.264%	8.727%
			0.314%	6.740%	0.063%	6.286%	2.378%	4.073%	5.721%	0.000%
			0.146%	6.111%	3.063%	1.542%	2.366%	1.477%	3.231%	6.768%
			0.203%	5.256%	1.320%	3.822%	2.435%	4.965%	2.940%	4.503%
		(0.1, 0.9)	1.057%	6.996%	2.016%	8.133%	2.232%	2.885%	0.789%	1.966%
			1.197%	0.861%	1.138%	2.288%	2.046%	2.527%	0.930%	0.015%
			1.363%	1.141%	2.087%	1.256%	2.681%	1.032%	1.309%	1.005%
			1.184%	0.400%	0.732%	6.372%	1.963%	0.771%	1.128%	1.730%
			0.635%	2.298%	2.658%	6.259%	1.635%	1.169%	1.062%	3.843%
			1.087%	2.339%	1.726%	4.862%	2.111%	1.677%	1.044%	1.712%
		(0.1, 1.9)	1.830%	3.599%	1.443%	3.655%	1.152%	0.000%	1.168%	0.808%
			0.995%	3.670%	1.833%	0.514%	1.437%	0.000%	1.259%	2.484%
			1.084%	0.000%	1.372%	0.000%	1.971%	0.555%	0.707%	4.208%
			1.548%	0.848%	0.853%	0.661%	0.465%	0.039%	0.493%	0.247%
			1.312%	0.154%	1.750%	0.204%	0.541%	0.343%	0.525%	0.117%
			1.354%	1.654%	1.450%	1.007%	1.113%	0.187%	0.830%	1.573%

(4,200)		Variable Cost	Machine Cost							
			(10, 12)		(10, 15)		(10, 20)		(10, 40)	
			(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*
Utilization	0.60	(0.1, 0.3)	0.027%	6.347%	2.167%	6.545%	0.067%	5.096%	1.678%	5.384%
			0.007%	0.525%	0.011%	1.214%	0.015%	0.472%	0.004%	0.000%
			0.052%	0.678%	0.005%	0.276%	0.611%	5.510%	0.035%	0.576%
			0.011%	0.552%	0.005%	0.028%	0.000%	0.000%	0.004%	0.000%
			0.002%	0.057%	0.124%	4.744%	0.028%	0.000%	0.000%	0.000%
			0.020%	1.632%	0.462%	2.561%	0.144%	2.216%	0.344%	1.192%
		(0.1, 0.9)	0.027%	0.134%	0.130%	3.579%	0.013%	0.000%	0.018%	0.227%
			0.011%	0.042%	0.078%	0.115%	1.123%	5.895%	0.036%	3.745%
			0.031%	0.323%	0.015%	0.012%	0.033%	0.256%	1.861%	0.000%
			0.000%	0.192%	0.007%	0.018%	0.001%	0.235%	0.031%	0.000%
			0.026%	0.069%	0.992%	2.240%	0.040%	0.151%	0.001%	0.000%
			0.019%	0.152%	0.244%	1.193%	0.242%	1.307%	0.389%	0.794%
		(0.1, 1.9)	0.032%	0.038%	0.040%	0.203%	0.039%	2.973%	0.005%	0.139%
			0.028%	0.092%	0.007%	0.181%	0.012%	0.410%	1.427%	0.205%
			0.014%	0.104%	0.066%	0.000%	0.010%	0.000%	0.261%	1.909%
			0.003%	0.000%	0.041%	0.022%	0.011%	0.000%	0.050%	0.000%
			0.042%	0.153%	0.014%	2.832%	0.845%	1.309%	0.030%	0.196%
			0.024%	0.077%	0.034%	0.648%	0.183%	0.938%	0.355%	0.490%
	0.75	(0.1, 0.3)	0.013%	0.159%	0.000%	0.000%	0.002%	0.449%	0.009%	0.000%
			0.024%	0.347%	0.001%	0.131%	0.000%	0.000%	0.000%	0.000%
			0.005%	0.000%	0.017%	8.447%	0.000%	0.000%	0.001%	0.091%
			0.000%	0.000%	0.000%	0.000%	0.000%	0.000%	0.000%	0.000%
			0.015%	0.683%	0.000%	0.000%	4.968%	0.000%	0.000%	0.000%
			0.011%	0.238%	0.004%	1.716%	0.994%	0.090%	0.002%	0.018%
		(0.1, 0.9)	0.019%	0.036%	0.166%	5.912%	0.069%	5.877%	0.008%	0.000%
			0.116%	6.416%	0.487%	5.129%	0.500%	5.313%	0.000%	0.000%
			0.024%	0.200%	0.258%	5.557%	0.011%	0.264%	0.000%	0.000%
			0.018%	0.000%	0.015%	0.000%	0.006%	0.000%	0.043%	4.432%
			0.348%	5.320%	0.016%	0.000%	3.812%	0.000%	0.050%	0.000%
			0.105%	2.394%	0.188%	3.320%	0.880%	2.291%	0.020%	0.886%
		(0.1, 1.9)	0.010%	0.000%	0.961%	2.747%	0.014%	0.000%	0.005%	0.000%
			0.158%	4.370%	2.887%	0.462%	0.808%	3.227%	0.768%	0.000%
			0.010%	0.000%	0.078%	4.018%	0.035%	3.900%	0.912%	0.662%
			0.103%	4.402%	0.021%	0.000%	0.146%	3.929%	0.164%	3.102%
			1.035%	2.848%	3.208%	0.359%	2.599%	0.000%	0.012%	0.155%
			0.263%	2.324%	1.431%	1.517%	0.720%	2.211%	0.372%	0.784%
	0.90	(0.1, 0.3)	2.974%	5.726%	1.640%	9.538%	0.000%	0.000%	4.583%	3.682%
			0.122%	0.266%	0.007%	1.112%	5.749%	0.000%	9.898%	0.000%
			0.003%	0.000%	0.000%	0.000%	3.954%	1.045%	6.406%	0.000%
			0.007%	0.108%	4.436%	6.533%	5.851%	2.073%	0.000%	0.000%
			2.255%	8.619%	0.412%	5.627%	7.585%	0.436%	10.603%	0.000%
			1.072%	2.944%	1.299%	4.562%	4.628%	0.711%	6.298%	0.736%
		(0.1, 0.9)	2.645%	7.203%	5.183%	1.998%	0.641%	8.078%	4.236%	2.703%
			3.157%	4.345%	0.290%	7.796%	0.000%	0.000%	4.154%	0.000%
			0.424%	8.885%	0.000%	0.000%	6.571%	0.000%	2.962%	0.000%
			0.358%	8.219%	6.977%	0.000%	5.251%	0.000%	0.000%	0.000%
			0.781%	7.328%	1.225%	6.991%	0.015%	0.000%	5.687%	0.000%
			1.473%	7.196%	2.735%	3.357%	2.496%	1.616%	3.408%	0.541%
		(0.1, 1.9)	2.688%	2.272%	1.483%	4.909%	1.993%	3.551%	4.547%	0.406%
			3.495%	8.225%	0.806%	5.210%	0.026%	5.567%	0.172%	0.000%
			4.205%	1.574%	0.215%	4.973%	4.625%	0.000%	2.006%	0.000%
			0.499%	4.531%	4.998%	0.118%	0.253%	4.777%	0.000%	0.000%
			1.451%	3.998%	4.257%	0.000%	4.441%	0.000%	0.508%	0.000%
			2.468%	4.120%	2.352%	3.042%	2.267%	2.779%	1.447%	0.081%

(15,92)		Variable Cost	Machine Cost							
			(10, 12)		(10, 15)		(10, 20)		(10, 40)	
			(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*
Utilization	0.60	(0.1, 0.3)	0.114%	0.000%	0.680%	3.120%	0.128%	3.290%	0.355%	0.000%
			0.047%	1.116%	0.195%	0.248%	0.292%	0.448%	0.202%	0.135%
			0.448%	1.708%	0.588%	0.170%	0.090%	0.352%	0.061%	1.583%
			0.140%	0.025%	0.497%	0.000%	0.017%	1.251%	0.077%	0.125%
			0.204%	1.082%	0.153%	0.207%	0.168%	2.195%	0.620%	0.000%
			0.191%	0.786%	0.423%	0.749%	0.139%	1.507%	0.263%	0.369%
		(0.1, 0.9)	0.341%	2.776%	0.100%	2.759%	0.156%	0.301%	0.023%	2.682%
			0.077%	1.499%	0.548%	0.348%	0.125%	0.000%	0.240%	0.481%
			0.233%	0.952%	0.172%	5.110%	0.178%	0.669%	0.106%	0.568%
			0.214%	1.486%	0.437%	0.742%	0.154%	0.982%	0.211%	0.000%
			0.174%	3.908%	0.189%	1.237%	0.202%	0.402%	0.179%	2.954%
			0.208%	2.124%	0.289%	2.039%	0.163%	0.471%	0.152%	1.337%
		(0.1, 1.9)	0.276%	2.817%	0.240%	2.405%	0.217%	2.013%	0.207%	0.000%
			0.173%	2.328%	0.189%	0.853%	0.237%	2.472%	0.078%	0.806%
			0.268%	3.434%	0.089%	0.873%	0.167%	4.320%	0.198%	0.590%
			0.176%	1.632%	0.284%	0.337%	0.147%	3.531%	0.286%	2.265%
			0.179%	0.710%	0.302%	0.213%	0.155%	2.827%	0.136%	3.370%
			0.215%	2.184%	0.221%	0.936%	0.185%	3.033%	0.181%	1.406%
	0.75	(0.1, 0.3)	0.145%	0.000%	0.128%	1.200%	1.315%	0.029%	0.117%	3.178%
			0.971%	1.357%	0.236%	0.263%	0.776%	0.754%	1.152%	0.000%
			0.677%	0.000%	1.202%	0.525%	1.320%	0.000%	1.261%	2.114%
			0.381%	4.069%	1.376%	0.396%	1.304%	7.857%	0.301%	0.374%
			1.359%	1.284%	0.916%	0.474%	1.233%	1.478%	1.095%	0.659%
			0.706%	1.342%	0.772%	0.571%	1.190%	2.024%	0.785%	1.265%
		(0.1, 0.9)	0.154%	2.367%	0.139%	3.188%	0.044%	2.353%	0.252%	0.474%
			0.108%	3.021%	0.379%	1.308%	0.179%	0.000%	0.342%	2.418%
			0.228%	8.465%	0.338%	1.082%	0.576%	2.700%	0.246%	0.000%
			0.417%	0.234%	0.538%	1.087%	0.576%	1.039%	0.193%	0.000%
			0.134%	0.000%	0.872%	0.835%	1.002%	1.197%	0.061%	1.554%
			0.208%	2.817%	0.453%	1.500%	0.475%	1.458%	0.219%	0.889%
		(0.1, 1.9)	0.308%	0.000%	0.315%	0.986%	0.111%	0.440%	0.548%	1.417%
			0.555%	4.917%	1.350%	2.695%	0.238%	7.229%	0.341%	0.276%
			0.992%	0.156%	0.548%	1.951%	0.966%	0.000%	0.872%	1.774%
			0.145%	0.465%	0.253%	2.648%	0.120%	0.161%	0.002%	0.000%
			0.354%	0.398%	0.234%	1.679%	0.308%	0.568%	0.036%	0.000%
			0.471%	1.187%	0.540%	1.992%	0.349%	1.680%	0.360%	0.693%
	0.90	(0.1, 0.3)	0.025%	0.348%	0.259%	0.000%	0.303%	2.625%	0.000%	0.000%
			0.078%	0.074%	0.000%	0.000%	0.167%	0.549%	1.214%	2.628%
			0.551%	5.912%	1.291%	1.114%	0.042%	0.000%	0.239%	1.885%
			0.459%	4.932%	0.729%	2.849%	1.139%	0.513%	2.153%	1.317%
			0.602%	5.307%	0.148%	0.000%	0.000%	0.000%	0.287%	0.000%
			0.343%	3.314%	0.485%	0.792%	0.330%	0.737%	0.779%	1.166%
		(0.1, 0.9)	0.054%	1.431%	1.710%	5.082%	2.602%	0.000%	0.369%	0.047%
			1.323%	4.017%	1.391%	0.658%	0.482%	3.252%	0.516%	0.000%
			1.656%	5.604%	0.594%	0.000%	0.722%	1.422%	0.577%	0.000%
			1.257%	7.184%	3.069%	0.000%	0.507%	1.607%	0.788%	0.000%
			0.345%	4.830%	1.061%	8.649%	0.455%	5.018%	0.326%	0.000%
			0.927%	4.613%	1.565%	2.878%	0.953%	2.260%	0.515%	0.009%
		(0.1, 1.9)	1.831%	0.040%	0.639%	2.322%	0.984%	0.000%	0.348%	0.081%
			0.499%	2.032%	0.342%	2.329%	0.292%	0.321%	0.012%	0.000%
			0.947%	1.063%	0.967%	1.039%	1.713%	0.000%	0.034%	0.000%
			1.198%	0.963%	0.575%	3.018%	0.802%	0.639%	0.016%	0.000%
			0.876%	0.089%	1.078%	0.700%	1.087%	0.023%	0.124%	1.001%
			1.070%	0.837%	0.720%	1.881%	0.976%	0.197%	0.107%	0.216%

(15,120)		Variable Cost	Machine Cost							
			(10, 12)		(10, 15)		(10, 20)		(10, 40)	
			(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*
Utilization	0.60	(0.1, 0.3)	0.123%	0.000%	0.193%	3.126%	0.069%	5.135%	0.062%	0.349%
			0.054%	3.246%	0.346%	0.336%	0.178%	0.274%	0.045%	0.000%
			0.069%	2.747%	0.188%	2.136%	0.123%	1.081%	0.717%	0.019%
			0.315%	0.969%	0.076%	0.000%	0.106%	1.854%	0.254%	0.209%
			0.024%	3.178%	0.024%	1.490%	0.340%	2.807%	0.203%	0.000%
			0.117%	2.028%	0.165%	1.418%	0.163%	2.230%	0.256%	0.115%
		(0.1, 0.9)	0.180%	1.237%	0.337%	0.906%	0.071%	0.777%	0.085%	0.029%
			0.110%	2.708%	0.188%	0.232%	0.155%	0.925%	0.341%	0.402%
			0.216%	2.003%	0.219%	0.253%	0.101%	0.016%	0.067%	0.932%
			0.098%	4.622%	0.299%	2.374%	0.146%	0.504%	0.072%	1.249%
			0.033%	1.111%	0.114%	1.254%	0.104%	0.663%	0.153%	0.300%
			0.127%	2.336%	0.232%	1.004%	0.115%	0.577%	0.143%	0.582%
		(0.1, 1.9)	0.330%	0.234%	0.124%	0.624%	0.513%	2.527%	0.052%	0.000%
			0.078%	1.626%	0.221%	1.251%	0.091%	2.487%	0.093%	0.231%
			0.225%	0.410%	0.115%	2.868%	0.111%	1.765%	0.175%	0.957%
			0.161%	0.529%	0.194%	1.366%	0.200%	0.802%	0.104%	0.393%
			0.115%	1.655%	0.160%	2.155%	0.239%	1.282%	0.061%	0.271%
			0.182%	0.891%	0.163%	1.653%	0.231%	1.772%	0.097%	0.370%
	0.75	(0.1, 0.3)	0.609%	0.773%	0.023%	2.042%	0.037%	1.397%	0.510%	0.000%
			0.786%	0.843%	0.335%	2.293%	0.475%	0.156%	0.220%	0.000%
			0.324%	0.780%	0.033%	0.879%	0.057%	0.464%	1.296%	0.436%
			0.380%	3.543%	0.501%	1.960%	0.165%	0.414%	0.069%	2.753%
			0.225%	1.750%	0.317%	0.786%	0.371%	0.000%	0.076%	0.333%
			0.465%	1.538%	0.242%	1.592%	0.221%	0.486%	0.434%	0.705%
		(0.1, 0.9)	0.138%	1.730%	0.260%	3.834%	0.204%	1.478%	0.074%	0.000%
			0.199%	2.462%	0.215%	1.263%	0.373%	1.288%	0.562%	0.000%
			0.199%	2.373%	0.202%	0.438%	0.134%	2.256%	0.142%	2.401%
			0.715%	0.786%	0.203%	1.809%	0.193%	3.967%	0.391%	0.134%
			0.209%	0.380%	0.195%	0.493%	0.605%	3.240%	0.266%	0.000%
			0.292%	1.546%	0.215%	1.567%	0.302%	2.446%	0.287%	0.507%
		(0.1, 1.9)	0.223%	0.582%	0.569%	0.225%	0.473%	0.000%	0.169%	1.211%
			0.459%	0.365%	0.225%	2.710%	0.102%	0.296%	0.256%	0.824%
			0.250%	4.013%	0.465%	0.676%	0.286%	0.801%	0.243%	0.189%
			0.494%	0.881%	0.192%	3.949%	0.285%	2.188%	0.125%	0.718%
			0.134%	1.954%	0.486%	0.873%	0.208%	0.545%	0.146%	0.266%
			0.312%	1.559%	0.387%	1.686%	0.271%	0.766%	0.188%	0.641%
	0.90	(0.1, 0.3)	0.498%	4.238%	0.479%	0.647%	0.823%	0.439%	2.850%	0.044%
			0.001%	0.000%	0.126%	4.675%	1.712%	2.590%	1.031%	1.988%
			2.413%	0.023%	0.003%	0.000%	1.390%	0.000%	0.599%	1.699%
			0.482%	1.229%	0.449%	4.190%	0.001%	4.844%	0.314%	0.494%
			1.815%	7.135%	0.008%	0.000%	2.614%	1.973%	1.506%	0.012%
			1.042%	2.525%	0.213%	1.902%	1.308%	1.969%	1.260%	0.848%
		(0.1, 0.9)	0.554%	0.000%	0.842%	0.656%	0.346%	2.822%	0.457%	0.058%
			0.305%	3.742%	0.510%	2.531%	2.303%	0.572%	0.134%	0.068%
			0.336%	4.086%	0.090%	4.223%	1.529%	0.000%	0.065%	2.204%
			0.443%	4.407%	1.772%	0.883%	0.300%	0.481%	0.115%	0.000%
			0.161%	0.283%	2.523%	1.483%	0.607%	1.540%	0.754%	0.163%
			0.360%	2.504%	1.147%	1.955%	1.017%	1.083%	0.305%	0.499%
		(0.1, 1.9)	0.630%	5.619%	0.905%	1.247%	0.536%	1.936%	0.328%	0.753%
			0.182%	3.355%	1.353%	0.842%	0.461%	1.392%	0.209%	3.132%
			0.359%	0.000%	0.462%	0.675%	1.360%	2.676%	0.076%	0.264%
			0.469%	0.327%	0.365%	0.000%	0.190%	0.000%	0.056%	0.000%
			0.581%	1.360%	0.713%	0.449%	0.063%	1.055%	0.013%	0.000%
			0.444%	2.132%	0.760%	0.643%	0.522%	1.412%	0.136%	0.830%

(15,200)	Variable Cost	Machine Cost							
Utilization		(10, 12)		(10, 15)		(10, 20)		(10, 40)	
		(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*
0.60	(0.1, 0.3)	0.026%	2.124%	0.079%	0.000%	0.184%	0.570%	0.424%	0.000%
		0.057%	0.134%	0.077%	0.586%	0.036%	1.715%	0.073%	0.024%
		0.386%	0.789%	0.160%	0.443%	0.156%	0.000%	0.121%	0.467%
		0.049%	0.103%	0.306%	2.219%	0.030%	0.539%	1.227%	0.037%
		0.366%	1.061%	0.078%	0.277%	0.041%	0.374%	0.000%	0.000%
		0.177%	0.842%	0.140%	0.705%	0.089%	0.640%	0.369%	0.106%
	(0.1, 0.9)	0.111%	0.181%	0.106%	0.282%	0.140%	0.475%	0.198%	0.151%
		0.049%	4.914%	0.078%	1.034%	0.094%	1.166%	0.129%	0.217%
		0.192%	0.537%	0.040%	0.563%	0.049%	2.188%	0.070%	0.588%
		0.116%	0.137%	0.100%	2.061%	0.338%	0.611%	0.035%	0.000%
		0.149%	2.605%	0.091%	1.043%	0.164%	0.555%	0.048%	0.150%
		0.123%	1.675%	0.083%	0.997%	0.157%	0.999%	0.096%	0.221%
	(0.1, 1.9)	0.166%	1.830%	0.104%	0.816%	0.151%	0.871%	0.119%	0.345%
		0.184%	1.422%	0.114%	1.967%	0.207%	0.334%	0.112%	0.542%
		0.162%	1.914%	0.107%	0.175%	0.172%	0.408%	0.142%	0.724%
		0.095%	1.186%	0.098%	1.298%	0.096%	0.494%	0.135%	0.273%
		0.262%	1.946%	0.106%	0.176%	0.183%	1.124%	0.072%	0.647%
		0.174%	1.660%	0.106%	0.886%	0.162%	0.646%	0.116%	0.506%
0.75	(0.1, 0.3)	0.167%	0.015%	0.000%	0.105%	0.028%	1.564%	0.180%	0.000%
		0.489%	2.274%	0.353%	2.378%	0.059%	5.441%	0.000%	0.000%
		0.024%	0.473%	0.448%	2.218%	0.162%	0.564%	1.342%	0.686%
		0.005%	3.433%	0.178%	0.405%	0.004%	0.035%	1.486%	0.147%
		0.100%	0.000%	0.396%	2.070%	0.214%	0.000%	0.652%	0.000%
		0.157%	1.239%	0.275%	1.435%	0.094%	1.521%	0.732%	0.167%
	(0.1, 0.9)	0.127%	1.673%	0.282%	3.128%	0.035%	1.718%	0.267%	0.887%
		0.167%	2.715%	0.087%	1.755%	0.227%	0.510%	0.384%	0.729%
		0.374%	0.330%	0.096%	0.672%	0.036%	2.051%	0.104%	0.112%
		0.062%	2.539%	0.182%	2.003%	0.063%	0.088%	0.753%	0.235%
		0.621%	0.403%	0.141%	0.769%	0.248%	1.479%	0.351%	1.313%
		0.270%	1.532%	0.158%	1.665%	0.122%	1.169%	0.372%	0.655%
	(0.1, 1.9)	0.226%	1.925%	0.269%	0.912%	0.268%	2.664%	0.083%	0.000%
		0.453%	0.672%	0.135%	1.796%	0.410%	0.545%	0.075%	1.015%
		0.215%	0.000%	0.246%	1.530%	0.228%	1.523%	0.140%	0.027%
		0.184%	1.233%	0.420%	0.692%	0.150%	0.198%	0.058%	0.826%
		0.543%	0.234%	0.271%	1.344%	0.114%	1.858%	0.082%	0.000%
		0.324%	0.813%	0.268%	1.255%	0.234%	1.358%	0.088%	0.374%
0.90	(0.1, 0.3)	0.035%	1.265%	0.002%	1.706%	0.002%	0.321%	0.401%	0.993%
		0.010%	0.808%	0.007%	0.878%	0.237%	0.000%	0.000%	0.000%
		0.015%	1.607%	0.012%	1.670%	0.683%	3.348%	1.754%	0.000%
		0.096%	1.760%	0.007%	0.329%	0.000%	0.000%	0.000%	0.000%
		0.000%	0.701%	0.000%	0.000%	0.301%	2.992%	0.059%	0.000%
		0.031%	1.228%	0.006%	0.917%	0.244%	1.332%	0.443%	0.199%
	(0.1, 0.9)	0.069%	1.691%	0.079%	0.703%	0.100%	0.716%	0.876%	0.869%
		0.084%	5.300%	0.055%	3.276%	0.003%	0.404%	1.844%	0.000%
		0.022%	1.023%	0.050%	5.141%	0.025%	0.077%	1.237%	1.919%
		0.033%	0.617%	0.182%	5.524%	1.228%	0.000%	0.837%	1.125%
		0.002%	0.293%	0.049%	0.681%	2.133%	0.560%	0.000%	0.000%
		0.042%	1.785%	0.083%	3.065%	0.698%	0.351%	0.959%	0.783%
	(0.1, 1.9)	0.001%	3.099%	0.205%	1.981%	0.162%	0.000%	0.031%	0.327%
		0.113%	0.293%	0.048%	0.000%	0.164%	0.000%	0.018%	0.000%
		0.006%	2.584%	0.256%	2.190%	0.018%	0.235%	0.033%	0.236%
		0.131%	0.200%	0.171%	3.693%	0.026%	1.901%	0.119%	0.030%
		0.023%	0.017%	0.354%	1.068%	0.074%	1.386%	0.138%	1.044%
		0.055%	1.238%	0.207%	1.786%	0.089%	0.704%	0.068%	0.327%

(40,400)		Variable Cost	Machine Cost (10, 12) (UB-LB) / LB	Machine Cost (10, 15) (UB-LB) / LB	Machine Cost (10, 20) (UB-LB) / LB	Machine Cost (10, 40) (UB-LB) / LB
Utilization	0.60	(0.1, 0.3)	1.382%	1.454%	2.019%	1.066%
			0.598%	0.424%	0.915%	0.203%
			0.497%	1.581%	1.595%	0.295%
			0.897%	0.138%	0.108%	0.245%
			0.798%	1.075%	0.796%	0.582%
			0.834%	0.934%	1.087%	0.478%
		(0.1, 0.9)	1.412%	0.503%	1.580%	0.540%
			1.369%	0.131%	0.596%	2.008%
			1.619%	1.555%	0.376%	0.021%
			0.837%	1.635%	0.321%	0.215%
			0.074%	0.128%	0.359%	0.043%
			1.062%	0.791%	0.646%	0.566%
		(0.1, 1.9)	1.614%	0.753%	1.260%	0.656%
			1.773%	1.398%	1.498%	0.638%
			1.398%	1.024%	0.725%	0.305%
			1.143%	2.074%	0.151%	0.198%
			1.397%	0.675%	0.489%	0.367%
			1.465%	1.185%	0.824%	0.433%
	0.75	(0.1, 0.3)	1.080%	1.063%	1.333%	0.645%
			0.015%	0.408%	0.869%	0.256%
			0.115%	1.006%	0.018%	0.267%
			0.036%	0.430%	1.665%	0.062%
			0.157%	1.272%	0.128%	0.264%
			0.281%	0.836%	0.803%	0.299%
		(0.1, 0.9)	0.656%	0.552%	0.917%	0.781%
			6.128%	3.457%	2.630%	0.219%
			1.667%	1.425%	2.976%	0.391%
			1.522%	1.253%	1.872%	0.782%
			0.944%	0.126%	0.374%	0.764%
			2.183%	1.363%	1.754%	0.587%
		(0.1, 1.9)	2.657%	0.815%	0.567%	1.256%
			1.520%	3.035%	0.621%	0.993%
			1.415%	4.781%	0.828%	0.513%
			1.191%	0.776%	1.642%	0.215%
			2.842%	1.403%	0.635%	0.707%
			1.925%	2.162%	0.859%	0.737%
	0.90	(0.1, 0.3)	3.222%	2.583%	2.498%	1.449%
			3.397%	2.630%	1.203%	1.242%
			2.001%	2.135%	0.779%	0.253%
			3.224%	2.222%	1.272%	0.517%
			2.981%	4.616%	2.131%	0.666%
			2.965%	2.837%	1.577%	0.825%
		(0.1, 0.9)	1.375%	1.906%	1.414%	0.246%
			1.987%	6.750%	0.884%	0.524%
			5.662%	7.248%	1.552%	0.858%
			1.570%	1.664%	2.937%	0.865%
			2.099%	0.988%	4.536%	0.638%
			2.538%	3.711%	2.265%	0.626%
		(0.1, 1.9)	1.309%	0.574%	0.162%	0.015%
			1.153%	0.998%	0.443%	0.737%
			0.946%	1.362%	3.033%	0.383%
			0.545%	0.971%	0.626%	0.345%
			6.427%	1.255%	2.125%	0.554%
			2.076%	1.032%	1.278%	0.407%

Means		Operating Cost	Procurement Cost							
			(10, 12)		(10, 15)		(10, 20)		(10, 40)	
			(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*	(Z*-LB) / LB	(UB-Z*) / Z*
Utilization	0.60	(0.1, 0.3)	0.092%	1.185%	0.230%	1.523%	0.241%	1.853%	0.687%	1.009%
		(0.1, 0.9)	0.082%	1.255%	0.138%	0.985%	0.139%	0.733%	0.260%	0.577%
		(0.1, 1.9)	0.137%	1.009%	0.094%	0.749%	0.174%	1.100%	0.157%	0.722%
	0.75	(0.1, 0.3)	0.235%	0.770%	0.302%	1.228%	0.563%	0.812%	0.567%	0.626%
		(0.1, 0.9)	0.167%	1.565%	0.206%	1.632%	0.380%	1.537%	0.513%	1.023%
		(0.1, 1.9)	0.248%	1.233%	0.463%	1.409%	0.297%	1.236%	0.486%	0.673%
	0.90	(0.1, 0.3)	0.452%	2.807%	0.557%	2.005%	1.520%	1.621%	1.955%	1.242%
		(0.1, 0.9)	0.655%	3.272%	1.217%	2.703%	1.226%	1.197%	1.042%	0.592%
		(0.1, 1.9)	0.722%	0.943%	1.381%	0.403%	1.036%	0.546%	0.226%	0.462%

APPENDIX B
EXPERIMENTAL STUDY RESULTS FOR MULTI-PERIOD CAPACITY ALLOCATION SOLUTION PROCEDURE

The results of the experimental analysis discussed in Chapter 3 are illustrated here. The numbers on the upper left corner of each table indicate number of periods, number of machine groups, and number of operations, respectively. GAP indicates the gap between the upper bound and the lower bound on the minimum value of the objective function. CPU time indicates the processing time of each problem. The last table displays overall means of ten problem structures.

(20,10,200)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	4	0.0982	16.36	0.0969	18.84	0.0842	19.61	0.0826	19.50	0.0752	19.77
		5	0.0830	18.89	0.1008	18.07	0.0822	21.04	0.0938	20.92	0.0765	20.76
		6	0.1028	18.68	0.1080	21.70	0.0909	22.25	0.0830	21.59	0.0819	22.14
		7	0.0783	23.29	0.1163	22.47	0.0898	23.01	0.0792	23.34	0.0928	23.39
		8	0.1084	21.86	0.1021	24.44	0.0975	24.50	0.0903	24.12	0.0891	25.05
	(10 , 14)	4	0.0905	17.03	0.0898	19.12	0.0815	17.75	0.0713	19.00	0.0690	19.83
		5	0.0926	18.45	0.0985	20.65	0.0906	18.84	0.0836	20.27	0.0717	20.27
		6	0.0937	19.44	0.1013	21.26	0.0730	20.71	0.0830	21.59	0.0853	21.37
		7	0.1007	20.93	0.1113	21.42	0.0938	22.84	0.0721	23.29	0.0845	23.56
		8	0.1058	20.55	0.0879	22.08	0.0909	24.61	0.0792	24.45	0.0891	24.99
	(10 , 17)	4	0.0890	14.99	0.0830	19.06	0.0706	17.96	0.0848	19.39	0.0621	20.27
		5	0.0900	17.85	0.0913	20.82	0.0839	19.45	0.0586	20.16	0.0748	21.09
		6	0.0931	20.93	0.0891	20.16	0.0870	21.20	0.0808	22.30	0.0684	22.08
		7	0.1047	20.16	0.1029	23.34	0.0771	22.74	0.0700	23.45	0.0643	22.57
		8	0.0967	22.46	0.1061	22.41	0.0859	23.89	0.0806	23.89	0.0724	24.82
	(10 , 20)	4	0.1120	15.48	0.0837	16.70	0.0741	18.34	0.0730	20.10	0.0709	19.17
		5	0.1104	16.53	0.0874	20.21	0.0782	20.32	0.0866	21.37	0.0808	21.15
		6	0.0954	18.24	0.0902	21.09	0.0908	21.09	0.0756	22.41	0.0732	22.41
		7	0.0894	19.93	0.0893	22.52	0.0825	22.41	0.0751	22.84	0.0763	22.68
		8	0.1079	23.78	0.0912	24.45	0.1049	25.21	0.0869	25.76	0.0822	24.88

(20,10,200)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	4	0.0726	16.10	0.0910	18.68	0.0827	19.00	0.0677	18.23	0.0721	18.07
		5	0.0962	15.66	0.0905	18.45	0.0793	20.27	0.0812	20.21	0.0697	19.88
		6	0.0816	17.90	0.0890	20.38	0.0824	20.93	0.0769	22.08	0.0836	22.25
		7	0.0809	20.98	0.0900	19.72	0.0794	21.81	0.0736	23.01	0.0805	23.23
		8	0.0892	21.03	0.1029	22.74	0.0934	24.17	0.0857	24.61	0.0801	24.99
	(10 , 14)	4	0.0716	16.31	0.0863	19.50	0.0749	17.19	0.0665	18.35	0.0695	19.44
		5	0.0884	17.63	0.0863	18.46	0.0759	19.45	0.0701	19.94	0.0686	20.81
		6	0.0890	18.18	0.0944	21.26	0.0742	19.78	0.0707	20.71	0.0726	19.88
		7	0.0824	22.79	0.0886	22.35	0.0852	22.69	0.0695	22.41	0.0847	24.05
		8	0.0873	23.56	0.0790	23.12	0.0896	24.88	0.0844	24.33	0.0773	24.72
	(10 , 17)	4	0.0735	15.00	0.0732	18.02	0.0637	18.40	0.0844	19.94	0.0585	19.45
		5	0.0750	17.80	0.0818	19.11	0.0729	19.22	0.0604	19.83	0.0698	20.65
		6	0.0972	19.33	0.0932	19.66	0.0846	21.69	0.0716	21.47	0.0659	20.10
		7	0.0905	19.72	0.0983	21.20	0.0780	21.97	0.0730	22.41	0.0695	22.52
		8	0.1060	19.99	0.0962	24.44	0.0961	24.28	0.0743	23.56	0.0667	24.01
	(10 , 20)	4	0.0860	15.71	0.0798	19.23	0.0638	18.84	0.0667	19.61	0.0638	20.26
		5	0.0944	17.79	0.0837	20.76	0.0723	19.94	0.0747	20.32	0.0596	20.04
		6	0.0806	20.87	0.0825	20.66	0.0862	21.04	0.0731	20.65	0.0631	21.53
		7	0.0863	20.76	0.0886	21.91	0.0751	22.85	0.0681	23.34	0.0683	22.57
		8	0.1013	23.72	0.0914	21.91	0.0976	24.16	0.0902	24.22	0.0819	25.16

(20,10,200)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	4	0.0829	16.04	0.1021	16.97	0.0839	19.06	0.0778	19.60	0.0722	19.55
		5	0.0675	20.65	0.0753	18.13	0.0811	21.37	0.0784	18.23	0.0676	20.26
		6	0.0791	16.20	0.0907	21.04	0.0657	19.89	0.0612	21.48	0.0737	21.86
		7	0.0737	20.04	0.0829	20.54	0.0744	22.03	0.0736	22.69	0.0666	22.25
		8	0.0813	22.63	0.1090	23.46	0.0931	24.00	0.0733	23.89	0.0632	24.34
	(10 , 14)	4	0.0820	16.31	0.0820	18.51	0.0723	18.95	0.0624	19.01	0.0671	19.06
		5	0.0870	17.30	0.0813	18.56	0.0660	19.50	0.0698	19.83	0.0644	20.32
		6	0.0691	20.65	0.0805	20.98	0.0634	19.72	0.0611	21.59	0.0657	21.15
		7	0.0927	18.67	0.0941	20.27	0.0751	22.13	0.0721	20.65	0.0687	23.18
		8	0.0992	20.92	0.0821	23.29	0.0888	25.10	0.0883	24.44	0.0818	24.50
	(10 , 17)	4	0.0706	17.69	0.0681	16.43	0.0631	18.23	0.0840	18.62	0.0538	19.50
		5	0.0777	19.06	0.0767	19.39	0.0636	19.66	0.0561	21.31	0.0570	21.04
		6	0.0792	21.20	0.0856	20.43	0.0726	20.92	0.0702	22.08	0.0635	21.58
		7	0.0910	19.28	0.0909	21.92	0.0663	22.41	0.0666	22.46	0.0628	21.75
		8	0.0940	21.75	0.0801	25.10	0.0859	24.34	0.0757	23.62	0.0676	25.21
	(10 , 20)	4	0.0788	17.14	0.0755	18.56	0.0633	19.27	0.0727	19.83	0.0656	19.55
		5	0.0898	16.70	0.0706	19.66	0.0696	21.36	0.0632	21.09	0.0734	21.20
		6	0.0744	19.78	0.0802	21.04	0.0693	20.54	0.0698	22.24	0.0622	22.30
		7	0.0754	21.09	0.0872	21.92	0.0809	22.41	0.0601	23.29	0.0717	22.96
		8	0.0962	21.42	0.0716	23.78	0.0997	23.90	0.0743	25.05	0.0775	25.38

(40,10,200)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	4	0.0393	52.18	0.0325	59.76	0.0310	53.72	0.0360	50.53	0.0276	60.37
		5	0.0459	54.43	0.0424	68.10	0.0410	63.66	0.0360	69.81	0.0327	69.26
		6	0.0767	77.94	0.0529	73.00	0.0440	72.67	0.0424	78.32	0.0385	68.66
		7	0.0876	77.78	0.0613	84.14	0.0586	84.31	0.0485	84.97	0.0438	81.56
		8	0.0854	90.25	0.0648	90.41	0.0518	88.37	0.0540	89.04	0.0473	95.46
	(10 , 14)	4	0.0391	50.42	0.0285	51.80	0.0199	56.69	0.0321	53.44	0.0323	57.78
		5	0.0529	60.53	0.0398	69.21	0.0423	57.78	0.0373	68.44	0.0441	69.26
		6	0.0662	70.86	0.0531	75.24	0.0506	76.12	0.0450	70.85	0.0388	80.47
		7	0.0748	74.92	0.0552	60.20	0.0566	82.00	0.0457	80.24	0.0479	85.36
		8	0.0923	78.93	0.0738	94.75	0.0616	89.96	0.0496	94.03	0.0463	92.66
	(10 , 17)	4	0.0748	46.58	0.0173	54.05	0.0254	52.72	0.0340	60.47	0.0254	52.12
		5	0.0865	49.26	0.0508	63.82	0.0316	61.24	0.0450	70.08	0.0447	74.09
		6	0.0786	72.94	0.0530	78.05	0.0439	66.46	0.0547	80.52	0.0436	71.74
		7	0.0986	72.17	0.0622	76.95	0.0594	79.20	0.0494	85.13	0.0509	84.64
		8	0.0979	82.66	0.0743	89.81	0.0610	96.34	0.0510	93.65	0.0528	92.99
	(10 , 20)	4	0.0315	54.10	0.0215	52.02	0.0289	51.95	0.0341	59.70	0.0376	59.54
		5	0.0557	47.40	0.0376	60.81	0.0455	62.51	0.0413	68.00	0.0344	71.90
		6	0.0682	71.30	0.0671	65.80	0.0447	78.87	0.0439	79.98	0.0469	64.54
		7	0.0895	81.29	0.0512	107.27	0.0550	100.46	0.0472	87.00	0.0437	82.94
		8	0.1045	70.69	0.0779	88.65	0.0660	109.19	0.0530	101.56	0.0440	98.92

(40,10,200)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	4	0.0485	51.63	0.0394	55.86	0.0322	63.61	0.0364	60.15	0.0328	65.25
		5	0.0594	70.80	0.0436	92.49	0.0484	84.09	0.0436	83.59	0.0344	68.82
		6	0.0829	74.53	0.0570	74.98	0.0526	78.00	0.0471	78.44	0.0468	67.89
		7	0.0932	82.00	0.0637	83.16	0.0540	80.36	0.0488	82.89	0.0467	87.77
		8	0.0872	87.44	0.0601	90.30	0.0581	84.04	0.0555	93.20	0.0507	86.89
	(10 , 14)	4	0.0590	49.21	0.0361	50.59	0.0320	49.32	0.0343	55.86	0.0315	56.57
		5	0.0597	61.73	0.0479	74.59	0.0460	73.33	0.0418	68.93	0.0494	69.10
		6	0.0779	69.59	0.0592	77.17	0.0508	74.04	0.0491	79.31	0.0428	79.69
		7	0.0778	75.31	0.0634	71.46	0.0558	76.29	0.0505	85.46	0.0480	86.94
		8	0.0867	68.17	0.0679	88.81	0.0593	91.67	0.0580	91.78	0.0483	96.23
	(10 , 17)	4	0.0699	47.62	0.0386	54.82	0.0343	56.14	0.0375	63.82	0.0331	52.62
		5	0.0857	54.98	0.0642	70.74	0.0367	61.19	0.0449	73.93	0.0512	69.92
		6	0.0786	77.78	0.0580	76.40	0.0541	70.52	0.0576	77.89	0.0503	77.34
		7	0.0984	74.70	0.0649	76.18	0.0601	80.41	0.0502	81.84	0.0501	87.60
		8	0.0841	73.38	0.0646	86.51	0.0593	90.46	0.0547	89.53	0.0481	93.92
	(10 , 20)	4	0.0527	45.53	0.0412	49.38	0.0344	57.18	0.0419	62.45	0.0367	64.98
		5	0.0638	53.77	0.0485	66.52	0.0512	67.07	0.0436	63.88	0.0374	70.96
		6	0.0907	62.89	0.0730	73.93	0.0512	74.70	0.0501	81.45	0.0528	75.85
		7	0.0882	77.17	0.0555	83.98	0.0562	85.58	0.0513	85.35	0.0465	81.24
		8	0.1009	84.69	0.0737	87.22	0.0664	95.13	0.0596	95.84	0.0491	91.18

(40,10,200)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	4	0.0505	62.45	0.0479	60.14	0.0404	72.67	0.0447	84.37	0.0451	67.17
		5	0.0630	67.12	0.0465	70.91	0.0526	70.47	0.0496	72.67	0.0421	68.44
		6	0.0781	120.62	0.0638	122.92	0.0524	77.72	0.0505	76.02	0.0471	77.50
		7	0.0861	79.75	0.0647	85.19	0.0510	83.65	0.0501	84.09	0.0462	80.41
		8	0.0752	91.23	0.0575	93.10	0.0522	92.05	0.0536	92.33	0.0453	95.24
	(10 , 14)	4	0.0461	59.26	0.0466	58.99	0.0408	65.36	0.0381	65.80	0.0357	60.47
		5	0.0658	62.18	0.0464	72.72	0.0511	71.85	0.0411	69.70	0.0536	67.88
		6	0.0837	62.07	0.0551	72.01	0.0601	75.80	0.0480	73.38	0.0490	78.54
		7	0.0692	72.83	0.0570	82.28	0.0638	85.58	0.0518	76.95	0.0428	80.69
		8	0.0828	87.88	0.0741	90.63	0.0565	89.92	0.0505	89.91	0.0475	90.13
	(10 , 17)	4	0.0632	55.14	0.0537	64.37	0.0464	61.74	0.0438	63.94	0.0394	64.59
		5	0.0965	55.69	0.0643	66.96	0.0464	63.00	0.0488	70.75	0.0554	62.23
		6	0.0834	69.15	0.0565	70.85	0.0533	73.06	0.0596	79.75	0.0523	80.08
		7	0.0765	77.83	0.0557	75.31	0.0550	83.54	0.0477	83.16	0.0454	83.38
		8	0.0857	81.23	0.0676	94.70	0.0531	91.23	0.0481	87.77	0.0474	89.26
	(10 , 20)	4	0.0451	52.95	0.0516	52.94	0.0401	59.71	0.0527	63.33	0.0483	66.13
		5	0.0755	57.84	0.0610	70.64	0.0614	68.87	0.0490	74.53	0.0423	74.15
		6	0.0868	70.63	0.0787	75.36	0.0624	75.63	0.0565	82.50	0.0546	74.98
		7	0.0732	76.41	0.0557	82.38	0.0567	85.35	0.0502	86.73	0.0412	81.07
		8	0.0972	83.65	0.0710	88.65	0.0701	94.80	0.0486	97.00	0.0454	92.82

(20,15,300)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	4	0.0567	42.07	0.0565	41.30	0.0542	42.84	0.0590	44.60	0.0504	45.97
		6	0.0547	43.83	0.0710	49.38	0.0642	47.57	0.0576	48.61	0.0532	48.88
		8	0.0685	48.28	0.0603	50.59	0.0639	51.52	0.0613	51.52	0.0666	52.45
		10	0.0757	52.57	0.0770	52.45	0.0696	54.76	0.0624	55.47	0.0570	55.09
		12	0.0851	58.88	0.0900	54.87	0.0819	59.54	0.0697	58.67	0.0707	59.81
	(10 , 14)	4	0.0533	39.11	0.0494	40.87	0.0540	44.10	0.0516	44.71	0.0470	45.15
		6	0.0583	45.48	0.0661	45.65	0.0561	46.91	0.0566	48.06	0.0559	47.56
		8	0.0736	46.57	0.0611	50.53	0.0647	51.41	0.0570	49.22	0.0486	50.25
		10	0.0721	53.66	0.0755	55.20	0.0620	55.86	0.0489	54.44	0.0587	56.19
		12	0.0815	54.76	0.0877	56.13	0.0828	57.84	0.0679	59.70	0.0622	60.58
	(10 , 17)	4	0.0714	34.99	0.0489	43.23	0.0661	41.47	0.0507	71.79	0.0491	55.14
		6	0.0657	43.89	0.0593	49.05	0.0596	47.67	0.0522	46.24	0.0513	46.64
		8	0.0704	56.52	0.0817	59.87	0.0590	52.95	0.0593	54.81	0.0549	53.66
		10	0.0778	55.70	0.0771	57.06	0.0707	56.07	0.0697	58.00	0.0559	59.65
		12	0.0864	59.75	0.0767	60.53	0.0669	62.12	0.0687	61.68	0.0660	63.00
	(10 , 20)	4	0.0627	43.39	0.0450	54.32	0.0560	72.40	0.0650	68.66	0.0512	48.94
		6	0.0636	42.41	0.0663	45.31	0.0504	48.67	0.0607	46.42	0.0660	48.66
		8	0.0735	49.44	0.0660	49.38	0.0616	48.83	0.0589	52.40	0.0513	51.08
		10	0.0761	54.99	0.0729	54.98	0.0662	55.04	0.0638	56.68	0.0604	55.70
		12	0.0903	56.79	0.0801	58.61	0.0642	59.76	0.0661	61.13	0.0639	60.31

(20,15,300)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	4	0.0579	39.82	0.0502	43.18	0.0496	43.22	0.0560	44.05	0.0537	44.33
		6	0.0591	43.83	0.0537	47.84	0.0584	46.63	0.0572	48.39	0.0497	46.47
		8	0.0573	48.50	0.0581	49.71	0.0575	50.36	0.0589	50.21	0.0566	51.14
		10	0.0708	48.45	0.0835	50.70	0.0637	55.36	0.0603	55.53	0.0560	55.42
		12	0.0776	57.89	0.0663	57.06	0.0728	58.56	0.0593	57.67	0.0625	59.87
	(10 , 14)	4	0.0531	41.03	0.0472	43.12	0.0558	43.23	0.0552	44.21	0.0478	43.50
		6	0.0620	46.41	0.0609	47.02	0.0633	46.08	0.0523	47.18	0.0517	49.04
		8	0.0659	52.34	0.0614	48.83	0.0660	50.04	0.0518	49.65	0.0433	49.87
		10	0.0707	52.73	0.0719	52.73	0.0655	53.55	0.0526	53.88	0.0497	56.18
		12	0.0737	57.34	0.0711	56.73	0.0682	58.49	0.0576	58.88	0.0589	60.47
	(10 , 17)	4	0.0630	38.28	0.0545	44.43	0.0632	42.46	0.0519	44.65	0.0509	45.21
		6	0.0711	46.80	0.0639	47.24	0.0601	46.96	0.0577	47.19	0.0544	46.41
		8	0.0687	51.80	0.0689	50.86	0.0593	51.52	0.0602	50.75	0.0480	50.92
		10	0.0713	54.60	0.0605	53.22	0.0608	55.48	0.0617	54.65	0.0549	54.76
		12	0.0817	56.30	0.0808	55.15	0.0657	58.99	0.0605	60.03	0.0605	61.08
	(10 , 20)	4	0.0629	40.48	0.0514	43.67	0.0573	43.94	0.0664	40.70	0.0507	40.81
		6	0.0647	40.54	0.0601	47.01	0.0534	48.17	0.0554	47.29	0.0588	48.01
		8	0.0765	42.07	0.0630	49.33	0.0568	48.45	0.0591	52.84	0.0510	49.71
		10	0.0676	52.95	0.0698	52.51	0.0599	54.05	0.0524	54.60	0.0528	54.87
		12	0.0872	52.95	0.0711	58.17	0.0624	58.71	0.0577	58.99	0.0669	61.46

(20,15,300)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	4	0.0428	38.17	0.0336	37.62	0.0372	39.88	0.0386	43.67	0.0392	44.32
		6	0.0498	39.54	0.0485	39.88	0.0427	43.28	0.0446	46.90	0.0387	47.73
		8	0.0462	45.86	0.0445	46.52	0.0457	45.97	0.0427	49.22	0.0459	49.98
		10	0.0466	45.21	0.0476	47.79	0.0451	66.68	0.0470	71.02	0.0384	52.35
		12	0.0493	57.45	0.0469	52.18	0.0491	55.81	0.0398	77.23	0.0435	106.72
	(10 , 14)	4	0.0524	45.92	0.0348	39.22	0.0419	42.85	0.0387	44.33	0.0320	44.00
		6	0.0434	43.66	0.0405	42.96	0.0448	44.43	0.0369	44.98	0.0409	44.00
		8	0.0572	48.94	0.0496	44.55	0.0486	49.21	0.0416	49.87	0.0357	47.51
		10	0.0511	46.25	0.0502	50.04	0.0453	53.38	0.0395	51.14	0.0422	53.01
		12	0.0496	56.30	0.0517	53.66	0.0439	51.42	0.0464	59.04	0.0485	58.38
	(10 , 17)	4	0.0540	35.75	0.0383	43.44	0.0448	38.67	0.0430	43.17	0.0387	41.69
		6	0.0466	41.52	0.0486	44.49	0.0430	40.86	0.0440	41.85	0.0368	43.83
		8	0.0452	48.17	0.0521	45.86	0.0425	45.75	0.0437	42.78	0.0414	47.13
		10	0.0437	47.78	0.0408	54.32	0.0519	53.17	0.0486	50.92	0.0443	50.14
		12	0.0528	54.76	0.0473	54.10	0.0492	56.90	0.0484	58.55	0.0403	58.93
	(10 , 20)	4	0.0726	32.85	0.0325	44.43	0.0351	42.29	0.0403	41.74	0.0358	42.46
		6	0.0504	38.94	0.0474	42.84	0.0457	43.94	0.0439	44.49	0.0439	46.25
		8	0.0459	48.99	0.0473	45.26	0.0436	45.86	0.0463	51.57	0.0369	49.48
		10	0.0524	50.91	0.0403	40.65	0.0454	52.73	0.0523	54.71	0.0395	52.73
		12	0.0545	52.01	0.0377	52.18	0.0437	58.88	0.0472	58.17	0.0479	53.82

(40,15,300)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	4	0.0417	98.76	0.0345	89.53	0.0286	96.56	0.0276	104.58	0.0304	106.55
		6	0.0373	102.60	0.0356	101.72	0.0273	110.56	0.0265	105.45	0.0246	100.02
		8	0.0403	111.83	0.0326	108.26	0.0315	117.38	0.0257	114.30	0.0313	128.31
		10	0.0420	124.35	0.0347	131.33	0.0314	137.26	0.0289	192.63	0.0295	236.02
		12	0.0489	135.73	0.0387	141.99	0.0358	147.81	0.0309	145.88	0.0309	151.48
	(10 , 14)	4	0.0420	100.90	0.0318	97.93	0.0349	98.53	0.0318	94.20	0.0272	96.01
		6	0.0383	115.84	0.0319	100.90	0.0266	103.48	0.0259	106.61	0.0278	103.43
		8	0.0444	115.07	0.0313	108.42	0.0283	125.34	0.0277	132.48	0.0260	106.61
		10	0.0502	122.09	0.0381	136.60	0.0351	121.11	0.0277	127.43	0.0293	149.29
		12	0.0546	137.53	0.0402	146.87	0.0333	146.49	0.0358	143.85	0.0314	159.56
	(10 , 17)	4	0.0555	84.70	0.0291	93.09	0.0379	98.64	0.0273	93.04	0.0261	102.93
		6	0.0381	105.46	0.0349	86.72	0.0340	109.96	0.0270	102.44	0.0241	114.57
		8	0.0467	114.85	0.0402	109.96	0.0329	112.21	0.0294	128.20	0.0259	120.18
		10	0.0573	130.55	0.0368	125.07	0.0330	125.17	0.0272	138.19	0.0260	141.99
		12	0.0541	142.15	0.0449	159.83	0.0368	160.11	0.0312	150.22	0.0280	164.39
	(10 , 20)	4	0.0518	151.93	0.0326	168.95	0.0324	94.58	0.0331	102.93	0.0263	93.16
		6	0.0478	97.38	0.0316	109.36	0.0314	107.05	0.0295	108.04	0.0279	105.01
		8	0.0398	112.21	0.0404	107.60	0.0320	113.53	0.0298	117.65	0.0279	119.19
		10	0.0563	125.39	0.0387	123.80	0.0320	131.60	0.0326	141.98	0.0312	140.17
		12	0.0481	170.82	0.0455	154.56	0.0381	148.96	0.0331	163.24	0.0307	145.99

(40,15,300)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	4	0.0305	106.33	0.0318	94.30	0.0266	97.88	0.0262	94.15	0.0255	98.15
		6	0.0353	105.18	0.0307	103.48	0.0297	99.36	0.0235	113.09	0.0249	96.89
		8	0.0410	116.12	0.0339	118.47	0.0293	119.57	0.0245	109.90	0.0263	127.64
		10	0.0411	132.21	0.0356	123.53	0.0325	132.59	0.0262	123.20	0.0268	122.05
		12	0.0507	154.99	0.0386	159.95	0.0319	140.44	0.0259	150.11	0.0297	144.01
	(10 , 14)	4	0.0352	92.77	0.0308	87.88	0.0356	100.51	0.0309	90.68	0.0249	96.61
		6	0.0389	102.44	0.0286	105.56	0.0281	99.86	0.0272	107.70	0.0265	184.11
		8	0.0382	178.56	0.0351	100.30	0.0287	122.60	0.0266	115.89	0.0223	115.23
		10	0.0461	121.93	0.0399	128.97	0.0319	131.06	0.0287	124.13	0.0248	141.21
		12	0.0523	143.36	0.0411	134.41	0.0307	132.64	0.0326	143.74	0.0298	147.15
	(10 , 17)	4	0.0478	93.32	0.0280	105.40	0.0333	96.40	0.0256	89.20	0.0229	90.74
		6	0.0352	101.50	0.0304	94.86	0.0286	103.59	0.0269	98.31	0.0252	102.16
		8	0.0431	110.46	0.0336	117.38	0.0308	118.64	0.0276	110.01	0.0240	112.87
		10	0.0533	128.19	0.0349	127.76	0.0335	142.47	0.0304	135.77	0.0275	142.70
		12	0.0466	146.60	0.0377	154.67	0.0392	140.12	0.0310	146.65	0.0256	150.00
	(10 , 20)	4	0.0435	102.88	0.0301	96.12	0.0360	100.30	0.0327	90.30	0.0238	90.52
		6	0.0442	96.61	0.0302	103.26	0.0293	107.82	0.0282	105.18	0.0253	100.95
		8	0.0404	115.89	0.0379	107.00	0.0284	111.93	0.0312	117.38	0.0260	123.04
		10	0.0480	121.50	0.0318	126.83	0.0310	137.09	0.0289	233.21	0.0293	138.36
		12	0.0389	143.63	0.0404	177.96	0.0340	151.70	0.0300	144.34	0.0278	145.33

(40,15,300)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	4	0.0263	108.15	0.0322	100.13	0.0227	96.56	0.0231	96.01	0.0262	103.64
		6	0.0278	120.17	0.0266	104.85	0.0215	110.01	0.0212	109.74	0.0210	112.32
		8	0.0331	110.68	0.0293	116.06	0.0280	120.95	0.0216	125.61	0.0253	134.02
		10	0.0393	125.89	0.0316	121.83	0.0299	134.24	0.0238	142.32	0.0242	126.44
		12	0.0426	175.16	0.0343	144.06	0.0279	136.71	0.0290	150.22	0.0235	145.33
	(10 , 14)	4	0.0290	85.91	0.0223	102.05	0.0297	96.34	0.0256	97.43	0.0259	94.81
		6	0.0289	114.58	0.0293	109.19	0.0231	106.72	0.0237	94.42	0.0206	109.08
		8	0.0303	103.32	0.0278	130.06	0.0278	111.67	0.0259	106.45	0.0194	116.99
		10	0.0388	132.26	0.0304	131.60	0.0313	138.52	0.0269	130.01	0.0261	136.60
		12	0.0421	164.99	0.0345	140.99	0.0281	167.58	0.0312	193.28	0.0290	252.27
	(10 , 17)	4	0.0385	109.63	0.0269	102.16	0.0303	95.62	0.0215	101.39	0.0265	86.35
		6	0.0367	83.37	0.0245	97.93	0.0271	105.90	0.0233	105.90	0.0231	95.85
		8	0.0360	124.90	0.0277	124.57	0.0267	111.55	0.0247	126.88	0.0229	102.00
		10	0.0507	149.34	0.0310	133.68	0.0308	127.71	0.0287	123.80	0.0234	133.36
		12	0.0462	144.51	0.0366	161.48	0.0320	149.29	0.0287	144.29	0.0229	158.57
	(10 , 20)	4	0.0368	109.68	0.0275	103.32	0.0270	96.83	0.0270	92.77	0.0216	87.49
		6	0.0373	101.62	0.0219	119.73	0.0248	103.98	0.0239	97.65	0.0198	103.86
		8	0.0276	114.46	0.0354	97.55	0.0231	115.39	0.0297	124.02	0.0230	121.55
		10	0.0371	154.72	0.0342	151.81	0.0292	127.87	0.0243	128.86	0.0278	151.21
		12	0.0468	164.67	0.0412	166.15	0.0280	155.06	0.0283	142.04	0.0257	139.95

(20,20,400)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	6	0.0580	61.96	0.0479	60.64	0.0525	65.03	0.0517	70.58	0.0559	75.52
		8	0.0608	64.10	0.0627	64.97	0.0518	74.04	0.0549	77.67	0.0546	73.05
		10	0.0656	74.54	0.0615	72.45	0.0671	78.33	0.0543	81.73	0.0584	81.79
		12	0.0680	72.78	0.0647	78.98	0.0671	82.55	0.0647	81.95	0.0554	85.08
		14	0.0761	90.90	0.0668	88.43	0.0643	91.89	0.0671	93.64	0.0587	90.96
		16	0.0832	98.26	0.0899	89.86	0.0740	89.08	0.0641	92.93	0.0648	93.59
	(10 , 14)	6	0.0689	58.99	0.0452	65.52	0.0553	60.41	0.0505	76.35	0.0509	72.34
		8	0.0586	63.94	0.0579	70.46	0.0485	79.92	0.0585	73.93	0.0524	79.31
		10	0.0766	71.46	0.0632	73.65	0.0589	80.85	0.0574	79.42	0.0536	78.55
		12	0.0699	83.37	0.0673	82.11	0.0619	83.54	0.0584	82.39	0.0533	81.07
		14	0.0754	80.41	0.0789	75.46	0.0644	93.82	0.0612	85.02	0.0605	93.31
		16	0.0853	87.22	0.0678	86.23	0.0683	94.26	0.0705	93.37	0.0642	97.82
	(10 , 17)	6	0.0661	59.20	0.0457	69.70	0.0460	67.23	0.0635	69.65	0.0435	73.38
		8	0.0669	65.31	0.0619	68.54	0.0527	71.95	0.0571	78.27	0.0518	71.90
		10	0.0599	80.63	0.0566	83.10	0.0556	75.42	0.0518	75.74	0.0541	81.12
		12	0.0837	70.63	0.0711	74.59	0.0610	88.05	0.0599	85.30	0.0617	81.73
		14	0.0836	90.19	0.0726	89.97	0.0672	79.43	0.0549	87.50	0.0597	96.07
		16	0.0780	89.80	0.0734	108.09	0.0711	102.27	0.0633	106.67	0.0680	96.78
	(10 , 20)	6	0.0446	66.68	0.0428	73.49	0.0460	67.17	0.0543	70.41	0.0442	66.67
		8	0.0478	74.97	0.0570	66.96	0.0492	72.06	0.0517	76.84	0.0511	81.46
		10	0.0539	74.10	0.0620	71.85	0.0507	82.77	0.0619	82.66	0.0649	82.16
		12	0.0507	65.25	0.0689	82.56	0.0647	84.48	0.0528	94.25	0.0560	104.69
		14	0.0738	96.88	0.0722	88.98	0.0614	93.32	0.0605	95.02	0.0597	97.87
		16	0.0752	105.73	0.0687	94.25	0.0581	102.33	0.0719	93.15	0.0603	97.11

(20,20,400)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	6	0.0527	56.57	0.0353	62.07	0.0440	59.05	0.0366	72.28	0.0359	72.39
		8	0.0495	63.66	0.0370	67.23	0.0310	68.99	0.0345	71.24	0.0347	69.92
		10	0.0481	75.85	0.0392	72.50	0.0398	79.04	0.0409	78.21	0.0390	86.34
		12	0.0648	92.50	0.0511	76.29	0.0421	75.97	0.0398	86.56	0.0436	75.36
		14	0.0776	71.51	0.0424	80.53	0.0502	88.15	0.0468	74.59	0.0413	86.56
		16	0.0710	75.69	0.0569	85.13	0.0495	87.11	0.0447	91.51	0.0471	95.07
	(10 , 14)	6	0.0572	58.77	0.0367	57.73	0.0378	62.83	0.0387	62.17	0.0419	68.60
		8	0.0503	65.42	0.0377	65.63	0.0331	76.68	0.0349	69.32	0.0368	67.89
		10	0.0685	67.83	0.0455	75.08	0.0454	85.74	0.0408	78.54	0.0384	72.12
		12	0.0849	69.15	0.0542	73.05	0.0479	71.08	0.0424	74.20	0.0430	81.02
		14	0.0607	85.08	0.0618	76.73	0.0427	91.01	0.0423	85.46	0.0457	79.53
		16	0.0945	76.40	0.0509	97.60	0.0552	92.16	0.0480	83.98	0.0465	98.21
	(10 , 17)	6	0.0650	55.64	0.0325	72.72	0.0424	55.70	0.0455	65.47	0.0352	68.05
		8	0.0821	61.96	0.0391	66.90	0.0443	63.54	0.0437	74.42	0.0328	71.19
		10	0.0550	74.92	0.0423	71.45	0.0412	78.76	0.0414	73.33	0.0415	80.46
		12	0.0723	66.07	0.0498	74.48	0.0479	82.94	0.0518	88.27	0.0424	81.78
		14	0.0688	73.93	0.0602	81.24	0.0493	92.33	0.0464	90.68	0.0463	83.00
		16	0.0635	82.55	0.0653	89.14	0.0537	93.10	0.0527	82.93	0.0562	87.99
	(10 , 20)	6	0.0326	63.66	0.0421	66.63	0.0380	67.33	0.0375	65.09	0.0421	62.73
		8	0.0439	72.61	0.0370	67.23	0.0415	76.13	0.0458	70.69	0.0358	72.77
		10	0.0398	82.06	0.0447	69.70	0.0429	76.90	0.0435	69.70	0.0393	76.73
		12	0.0400	71.02	0.0507	96.50	0.0515	100.63	0.0393	94.47	0.0473	81.62
		14	0.0439	88.04	0.0574	84.20	0.0582	84.37	0.0500	91.62	0.0452	90.13
		16	0.0645	86.56	0.0589	88.98	0.0500	94.69	0.0567	90.79	0.0514	86.40

(20,20,400)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	6	0.0684	48.23	0.0300	49.00	0.0392	54.04	0.0327	63.72	0.0317	71.34
		8	0.0832	51.08	0.0442	47.13	0.0301	75.25	0.0294	78.66	0.0286	77.55
		10	0.0900	69.70	0.0375	74.97	0.0359	95.57	0.0346	73.54	0.0350	75.74
		12	0.0906	56.57	0.0454	77.17	0.0418	79.26	0.0330	80.52	0.0378	79.53
		14	0.0840	61.47	0.0532	63.49	0.0414	69.76	0.0384	87.99	0.0379	83.98
		16	0.0692	78.21	0.0547	82.61	0.0383	90.14	0.0349	76.35	0.0389	96.34
	(10 , 14)	6	0.0330	65.03	0.0382	53.66	0.0309	59.54	0.0267	76.13	0.0286	71.40
		8	0.0362	58.66	0.0512	57.24	0.0263	69.65	0.0283	70.80	0.0348	73.92
		10	0.0975	45.09	0.0413	66.29	0.0352	78.82	0.0353	72.66	0.0302	63.72
		12	0.0726	62.84	0.0352	81.23	0.0357	75.25	0.0357	73.38	0.0337	77.39
		14	0.0673	69.87	0.0538	72.62	0.0388	82.60	0.0366	91.78	0.0335	80.36
		16	0.0930	63.16	0.0421	83.32	0.0454	89.14	0.0360	82.22	0.0365	88.60
	(10 , 17)	6	0.0715	48.12	0.0217	61.85	0.0351	50.36	0.0271	64.54	0.0289	56.25
		8	0.0676	57.45	0.0299	68.49	0.0334	59.43	0.0347	73.43	0.0288	75.53
		10	0.0575	57.61	0.0309	72.34	0.0359	73.76	0.0331	70.14	0.0339	86.57
		12	0.0615	65.47	0.0354	72.78	0.0380	77.72	0.0360	81.84	0.0299	86.95
		14	0.0531	69.59	0.0370	81.13	0.0376	77.72	0.0398	79.92	0.0312	88.43
		16	0.0502	83.65	0.0426	93.49	0.0414	86.78	0.0403	88.65	0.0493	87.22
	(10 , 20)	6	0.0309	62.50	0.0249	65.80	0.0339	56.79	0.0291	75.91	0.0244	69.87
		8	0.0278	77.50	0.0374	56.13	0.0355	58.44	0.0280	76.68	0.0352	69.05
		10	0.0389	75.41	0.0330	71.90	0.0360	76.56	0.0332	79.69	0.0294	70.58
		12	0.0296	51.52	0.0352	74.04	0.0426	101.78	0.0351	99.03	0.0373	91.45
		14	0.0455	66.24	0.0377	77.56	0.0409	87.82	0.0401	78.00	0.0299	78.98
		16	0.0401	87.06	0.0459	93.48	0.0417	97.82	0.0437	83.43	0.0373	91.73

(40,20,400)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	6	0.0784	148.52	0.0665	162.86	0.0493	170.33	0.0592	164.01	0.0495	171.20
		8	0.0819	176.36	0.0676	187.40	0.0531	183.78	0.0441	179.00	0.0508	181.80
		10	0.0565	191.47	0.0530	211.47	0.0505	216.35	0.0501	221.78	0.0506	230.13
		12	0.0628	202.68	0.0649	210.85	0.0607	225.03	0.0493	229.75	0.0497	219.81
		14	0.0798	253.81	0.0743	246.18	0.0588	260.07	0.0579	262.05	0.0510	267.60
		16	0.0924	280.40	0.0669	305.55	0.0595	271.72	0.0505	293.63	0.0497	300.33
	(10 , 14)	6	0.0589	152.92	0.0627	198.67	0.0519	182.47	0.0517	163.24	0.0505	176.91
		8	0.0546	165.10	0.0498	179.66	0.0442	179.94	0.0552	197.63	0.0410	193.07
		10	0.0751	188.94	0.0553	240.47	0.0451	227.28	0.0488	213.50	0.0447	229.53
		12	0.0635	208.78	0.0570	217.72	0.0464	226.02	0.0456	242.11	0.0465	222.56
		14	0.0688	231.07	0.0533	240.30	0.0575	260.40	0.0600	243.54	0.0511	257.38
		16	0.0820	245.79	0.0626	256.23	0.0567	303.19	0.0536	266.44	0.0445	290.78
	(10 , 17)	6	0.0932	154.34	0.0608	170.77	0.0475	162.03	0.0661	158.02	0.0488	180.38
		8	0.0877	158.02	0.0522	191.14	0.0480	177.63	0.0508	202.02	0.0447	186.58
		10	0.0903	203.34	0.0531	207.84	0.0529	196.09	0.0452	197.07	0.0453	256.94
		12	0.0722	193.12	0.0490	216.30	0.0557	261.72	0.0457	247.93	0.0472	239.97
		14	0.0906	221.29	0.0652	265.18	0.0544	286.82	0.0493	303.14	0.0506	240.96
		16	0.0754	256.50	0.0601	257.87	0.0613	279.68	0.0572	283.91	0.0479	285.61
	(10 , 20)	6	0.0476	170.00	0.0711	175.43	0.0503	179.33	0.0560	162.96	0.0566	168.79
		8	0.0716	179.28	0.0698	189.55	0.0438	182.79	0.0447	184.82	0.0531	183.01
		10	0.0571	189.72	0.0600	200.64	0.0501	223.54	0.0470	209.82	0.0529	209.87
		12	0.0638	219.48	0.0624	219.37	0.0555	247.50	0.0508	220.69	0.0518	244.31
		14	0.0625	245.74	0.0616	240.02	0.0540	273.26	0.0577	289.73	0.0470	243.97
		16	0.0718	274.19	0.0654	268.59	0.0554	275.67	0.0608	288.19	0.0507	322.85

(40,20,400)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	6	0.0489	181.80	0.0386	166.92	0.0418	167.25	0.0462	164.88	0.0391	163.90
		8	0.0547	171.70	0.0383	173.83	0.0384	177.90	0.0352	169.72	0.0375	174.33
		10	0.0520	181.31	0.0456	198.67	0.0403	187.85	0.0402	206.85	0.0404	203.22
		12	0.0502	211.80	0.0397	207.34	0.0377	201.58	0.0442	229.10	0.0380	219.04
		14	0.0720	222.99	0.0512	239.91	0.0471	231.95	0.0400	233.77	0.0382	239.64
		16	0.0573	254.47	0.0488	259.36	0.0403	271.55	0.0371	287.53	0.0426	285.18
	(10 , 14)	6	0.0430	170.87	0.0473	178.89	0.0469	173.23	0.0410	184.28	0.0439	196.47
		8	0.0493	185.98	0.0382	170.21	0.0395	181.31	0.0426	196.53	0.0341	174.94
		10	0.0691	197.23	0.0482	181.36	0.0377	193.29	0.0408	242.33	0.0374	223.60
		12	0.0539	235.35	0.0499	229.59	0.0411	212.89	0.0376	236.12	0.0425	219.81
		14	0.0548	249.69	0.0453	234.09	0.0522	243.60	0.0450	215.52	0.0361	247.93
		16	0.0593	264.03	0.0536	276.55	0.0441	242.06	0.0421	265.90	0.0377	260.12
	(10 , 17)	6	0.0865	150.77	0.0484	157.97	0.0418	168.90	0.0537	175.16	0.0395	165.00
		8	0.0958	174.50	0.0435	179.55	0.0440	196.03	0.0418	180.81	0.0388	176.92
		10	0.0632	197.90	0.0438	199.60	0.0430	209.37	0.0375	202.95	0.0405	236.13
		12	0.0788	196.90	0.0486	260.51	0.0467	213.17	0.0405	232.06	0.0396	226.13
		14	0.0710	228.49	0.0490	229.20	0.0503	237.22	0.0425	256.78	0.0417	266.39
		16	0.0554	275.17	0.0488	283.75	0.0542	282.53	0.0435	264.96	0.0426	285.28
	(10 , 20)	6	0.0428	159.01	0.0527	166.87	0.0433	178.68	0.0432	179.99	0.0525	178.12
		8	0.0474	189.44	0.0654	185.15	0.0411	179.55	0.0387	183.73	0.0407	192.19
		10	0.0475	184.88	0.0443	201.91	0.0401	208.71	0.0399	206.14	0.0445	197.34
		12	0.0463	215.69	0.0595	218.72	0.0498	228.49	0.0458	232.83	0.0423	246.12
		14	0.0507	246.78	0.0530	245.02	0.0476	244.31	0.0458	264.14	0.0396	268.86
		16	0.0550	303.74	0.0603	238.11	0.0470	256.55	0.0530	260.78	0.0441	265.07

(40,20,400)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	6	0.0461	157.91	0.0394	156.60	0.0377	165.66	0.0382	175.00	0.0366	203.33
		8	0.0474	183.12	0.0407	179.89	0.0326	193.44	0.0355	186.25	0.0344	174.00
		10	0.0430	201.74	0.0376	183.18	0.0390	203.60	0.0374	204.65	0.0343	205.48
		12	0.0447	219.21	0.0427	212.89	0.0395	216.57	0.0401	209.32	0.0370	209.05
		14	0.0493	235.13	0.0434	228.76	0.0382	248.10	0.0383	242.38	0.0345	234.26
		16	0.0528	255.68	0.0454	263.20	0.0349	265.23	0.0368	260.35	0.0379	288.42
	(10 , 14)	6	0.0407	170.82	0.0424	158.41	0.0395	162.36	0.0351	169.23	0.0392	174.34
		8	0.0439	178.13	0.0360	188.28	0.0376	196.85	0.0382	209.98	0.0340	191.74
		10	0.0475	189.71	0.0385	204.37	0.0351	214.49	0.0349	201.69	0.0383	201.52
		12	0.0505	205.15	0.0418	248.59	0.0404	218.77	0.0351	211.24	0.0387	242.00
		14	0.0495	221.51	0.0438	250.24	0.0396	226.95	0.0374	212.56	0.0370	231.51
		16	0.0560	260.57	0.0471	270.45	0.0402	260.13	0.0390	254.75	0.0368	245.35
	(10 , 17)	6	0.0615	166.86	0.0454	151.65	0.0394	164.34	0.0463	157.09	0.0354	168.52
		8	0.0682	165.27	0.0391	176.81	0.0366	183.18	0.0379	182.79	0.0355	183.23
		10	0.0483	192.51	0.0458	188.28	0.0369	187.79	0.0369	205.31	0.0372	194.38
		12	0.0799	209.76	0.0425	215.20	0.0436	220.36	0.0389	213.61	0.0363	234.15
		14	0.0647	253.26	0.0446	239.80	0.0455	225.63	0.0371	228.49	0.0387	262.98
		16	0.0465	244.75	0.0426	237.17	0.0454	307.69	0.0448	268.31	0.0343	258.21
	(10 , 20)	6	0.0414	155.77	0.0512	158.30	0.0420	165.33	0.0423	160.11	0.0426	173.79
		8	0.0419	165.49	0.0419	178.18	0.0407	178.73	0.0384	185.70	0.0399	202.51
		10	0.0411	199.05	0.0465	185.59	0.0350	208.50	0.0354	208.33	0.0366	190.31
		12	0.0526	208.23	0.0464	232.77	0.0426	229.31	0.0414	204.87	0.0414	245.68
		14	0.0420	238.60	0.0454	260.24	0.0448	264.02	0.0413	266.88	0.0368	244.09
		16	0.0482	257.54	0.0487	254.63	0.0421	285.94	0.0441	271.55	0.0390	329.94

(20,25,500)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	6	0.0580	100.24	0.0536	110.01	0.0470	106.99	0.0502	110.29	0.0399	117.70
		9	0.0593	116.94	0.0512	114.08	0.0548	117.87	0.0466	118.96	0.0393	125.72
		12	0.0633	130.89	0.0575	117.10	0.0545	117.49	0.0585	124.90	0.0552	125.01
		15	0.0775	139.07	0.0787	126.27	0.0618	128.36	0.0614	131.05	0.0568	134.40
		18	0.0724	157.74	0.0788	125.06	0.0804	145.78	0.0566	151.26	0.0590	141.88
		21	0.0913	149.78	0.0935	145.33	0.0907	154.94	0.0725	165.33	0.0698	158.19
	(10 , 14)	6	0.0544	101.34	0.0468	101.83	0.0491	109.03	0.0430	115.61	0.0472	115.67
		9	0.0529	107.48	0.0492	108.09	0.0489	120.84	0.0499	125.62	0.0496	127.16
		12	0.0731	127.92	0.0655	149.18	0.0559	147.20	0.0543	141.87	0.0561	129.08
		15	0.0791	138.63	0.0635	141.76	0.0723	143.08	0.0589	139.78	0.0554	140.28
		18	0.0811	138.25	0.0688	145.94	0.0729	143.08	0.0660	146.65	0.0621	148.02
		21	0.0900	123.86	0.0856	150.71	0.0742	174.94	0.0712	173.51	0.0674	175.38
	(10 , 17)	6	0.0470	110.18	0.0414	116.67	0.0454	112.60	0.0418	119.41	0.0462	122.81
		9	0.0513	122.87	0.0514	120.28	0.0440	122.21	0.0510	126.60	0.0488	129.85
		12	0.0631	132.10	0.0546	132.81	0.0506	137.58	0.0539	143.35	0.0563	135.72
		15	0.0752	150.44	0.0724	126.05	0.0560	137.31	0.0564	157.58	0.0530	136.49
		18	0.0813	145.39	0.0715	136.60	0.0682	144.13	0.0673	149.67	0.0586	151.81
		21	0.0935	130.67	0.0832	145.22	0.0778	159.72	0.0745	164.39	0.0691	159.67
	(10 , 20)	6	0.0458	94.26	0.0503	113.47	0.0460	117.05	0.0400	120.06	0.0468	115.84
		9	0.0560	121.33	0.0553	121.22	0.0546	119.85	0.0460	120.34	0.0450	123.14
		12	0.0494	119.57	0.0522	121.22	0.0533	129.79	0.0536	136.38	0.0475	131.16
		15	0.0730	114.69	0.0619	136.71	0.0592	133.97	0.0535	133.47	0.0635	143.85
		18	0.0817	132.92	0.0683	144.84	0.0664	142.86	0.0655	149.13	0.0557	145.00
		21	0.0883	156.48	0.0916	154.01	0.0763	160.11	0.0677	165.54	0.0626	165.32

(20,25,500)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	6	0.0425	94.20	0.0355	84.92	0.0320	91.34	0.0300	97.66	0.0268	97.76
		9	0.0421	111.06	0.0387	104.19	0.0314	120.07	0.0333	116.39	0.0255	105.62
		12	0.0658	98.75	0.0513	98.15	0.0351	137.64	0.0393	138.14	0.0337	138.57
		15	0.0731	120.23	0.0536	110.29	0.0436	112.82	0.0477	120.50	0.0406	137.92
		18	0.0700	117.65	0.0547	129.12	0.0542	123.75	0.0468	145.28	0.0436	140.50
		21	0.0701	109.19	0.0669	124.51	0.0572	148.19	0.0517	160.71	0.0500	151.38
	(10 , 14)	6	0.0457	112.05	0.0304	112.65	0.0305	107.38	0.0304	118.20	0.0278	101.06
		9	0.0494	103.10	0.0421	98.48	0.0339	103.26	0.0328	111.77	0.0320	111.72
		12	0.0499	115.84	0.0420	95.52	0.0416	113.26	0.0368	120.34	0.0381	124.46
		15	0.0689	105.18	0.0561	111.06	0.0473	121.28	0.0397	129.40	0.0372	133.96
		18	0.0786	116.77	0.0553	126.22	0.0497	148.25	0.0452	128.80	0.0465	140.94
		21	0.0921	128.14	0.0562	146.60	0.0535	156.48	0.0482	142.59	0.0537	146.92
	(10 , 17)	6	0.0674	90.14	0.0324	110.07	0.0362	108.48	0.0290	117.11	0.0286	105.62
		9	0.0380	104.53	0.0395	83.27	0.0325	116.44	0.0381	100.13	0.0385	121.38
		12	0.0402	123.91	0.0414	109.91	0.0416	125.01	0.0409	112.93	0.0362	120.89
		15	0.0552	122.37	0.0521	123.19	0.0432	125.89	0.0367	135.28	0.0387	123.91
		18	0.0632	122.54	0.0558	138.52	0.0548	130.89	0.0459	122.71	0.0486	150.94
		21	0.0687	156.75	0.0540	144.79	0.0491	154.23	0.0487	149.45	0.0590	171.97
	(10 , 20)	6	0.0309	124.30	0.0486	109.30	0.0358	119.74	0.0343	116.44	0.0340	114.08
		9	0.0277	113.92	0.0443	114.13	0.0347	113.75	0.0337	116.33	0.0342	118.03
		12	0.0656	100.68	0.0357	108.76	0.0353	112.60	0.0383	127.26	0.0384	126.44
		15	0.0507	132.65	0.0503	112.43	0.0395	137.76	0.0419	167.69	0.0430	140.55
		18	0.0515	154.73	0.0511	147.86	0.0505	132.54	0.0472	149.62	0.0436	134.63
		21	0.0587	137.86	0.0654	160.10	0.0494	158.95	0.0539	163.18	0.0462	152.53

(20,25,500)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	6	0.0299	72.01	0.0297	70.30	0.0238	73.44	0.0222	108.15	0.0223	104.13
		9	0.0345	82.61	0.0283	79.03	0.0267	80.41	0.0275	70.91	0.0219	64.09
		12	0.0368	75.74	0.0290	79.15	0.0270	80.08	0.0253	75.30	0.0230	122.65
		15	0.0419	92.99	0.0395	94.64	0.0295	87.33	0.0296	94.52	0.0263	120.83
		18	0.0402	96.50	0.0358	102.22	0.0321	89.25	0.0317	143.03	0.0348	92.49
		21	0.0536	108.58	0.0436	101.67	0.0416	120.73	0.0372	166.64	0.0312	153.13
	(10 , 14)	6	0.0325	69.76	0.0362	66.68	0.0270	103.48	0.0260	62.01	0.0227	111.83
		9	0.0365	72.61	0.0327	76.45	0.0214	113.21	0.0205	110.40	0.0221	77.66
		12	0.0355	77.50	0.0278	83.10	0.0241	120.29	0.0220	121.33	0.0193	128.20
		15	0.0512	89.36	0.0345	125.50	0.0297	121.55	0.0279	135.61	0.0248	134.18
		18	0.0768	105.24	0.0318	130.07	0.0354	130.29	0.0325	112.82	0.0338	91.89
		21	0.0595	123.36	0.0431	158.52	0.0351	112.66	0.0366	150.61	0.0322	155.00
	(10 , 17)	6	0.0328	98.65	0.0222	102.11	0.0228	106.07	0.0208	113.31	0.0226	105.13
		9	0.0228	108.37	0.0218	96.67	0.0240	114.14	0.0194	73.76	0.0200	112.92
		12	0.0293	87.66	0.0254	123.85	0.0253	120.46	0.0262	151.87	0.0223	127.54
		15	0.0438	96.67	0.0344	141.77	0.0308	129.19	0.0280	137.87	0.0222	99.47
		18	0.0293	137.25	0.0420	143.85	0.0295	143.52	0.0341	139.90	0.0304	144.01
		21	0.0433	144.78	0.0331	156.04	0.0361	159.01	0.0333	153.74	0.0277	161.04
	(10 , 20)	6	0.0240	103.86	0.0321	72.50	0.0298	114.91	0.0254	116.28	0.0297	115.73
		9	0.0200	112.59	0.0270	113.47	0.0243	118.97	0.0220	114.24	0.0206	116.50
		12	0.0509	88.21	0.0227	122.49	0.0308	113.92	0.0304	132.81	0.0270	126.11
		15	0.0312	117.15	0.0362	123.42	0.0263	135.61	0.0308	131.87	0.0269	133.96
		18	0.0374	133.97	0.0305	135.61	0.0320	139.40	0.0309	140.50	0.0277	134.95
		21	0.0446	150.94	0.0488	155.44	0.0427	163.40	0.0290	152.64	0.0330	147.91

(40,25,500)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	6	0.0343	255.96	0.0370	240.58	0.0346	245.13	0.0326	254.96	0.0319	242.77
		9	0.0439	273.31	0.0377	259.36	0.0360	289.46	0.0336	295.72	0.0357	270.67
		12	0.0563	310.21	0.0440	295.61	0.0420	325.10	0.0363	373.83	0.0351	329.39
		15	0.0490	338.56	0.0475	341.25	0.0429	336.64	0.0375	345.15	0.0373	372.67
		18	0.0529	395.96	0.0488	381.12	0.0506	381.29	0.0415	404.14	0.0395	411.56
		21	0.0580	427.48	0.0511	435.18	0.0456	443.85	0.0474	395.80	0.0433	440.34
	(10 , 14)	6	0.0503	220.70	0.0389	265.51	0.0358	261.12	0.0332	261.72	0.0384	256.17
		9	0.0455	260.23	0.0370	306.32	0.0395	249.09	0.0374	294.73	0.0347	270.89
		12	0.0550	360.98	0.0428	310.93	0.0394	300.11	0.0396	314.23	0.0389	283.69
		15	0.0568	354.44	0.0460	381.51	0.0472	396.68	0.0357	388.66	0.0383	379.42
		18	0.0669	393.32	0.0463	398.70	0.0498	395.51	0.0424	410.34	0.0379	353.72
		21	0.0702	424.85	0.0557	451.54	0.0458	519.87	0.0410	444.34	0.0444	442.92
	(10 , 17)	6	0.0454	252.11	0.0339	271.06	0.0446	242.38	0.0397	250.68	0.0349	259.58
		9	0.0414	285.89	0.0360	308.03	0.0417	323.67	0.0395	269.08	0.0340	283.47
		12	0.0481	305.82	0.0431	318.63	0.0426	304.61	0.0399	293.57	0.0377	335.98
		15	0.0555	352.90	0.0513	336.91	0.0464	328.84	0.0381	337.62	0.0386	360.75
		18	0.0643	384.25	0.0575	360.59	0.0473	390.80	0.0400	413.37	0.0426	434.03
		21	0.0616	458.63	0.0578	443.96	0.0566	423.59	0.0491	417.32	0.0505	438.36
	(10 , 20)	6	0.0556	242.77	0.0376	274.03	0.0370	268.97	0.0337	287.26	0.0387	314.23
		9	0.0411	282.81	0.0414	271.06	0.0444	264.19	0.0371	276.60	0.0356	268.20
		12	0.0438	302.15	0.0429	396.94	0.0428	311.87	0.0399	302.20	0.0352	295.44
		15	0.0707	322.47	0.0470	366.19	0.0472	361.91	0.0393	332.90	0.0446	358.66
		18	0.0491	413.42	0.0515	394.31	0.0491	387.33	0.0445	408.04	0.0458	421.77
		21	0.0706	423.31	0.0579	468.02	0.0538	465.49	0.0489	453.90	0.0419	452.03

(40,25,500)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	6	0.0349	282.04	0.0354	270.34	0.0335	253.04	0.0320	251.34	0.0329	243.65
		9	0.0419	269.84	0.0368	266.45	0.0340	279.08	0.0330	325.16	0.0322	291.82
		12	0.0437	304.51	0.0398	276.22	0.0373	288.97	0.0334	299.68	0.0319	265.24
		15	0.0452	344.99	0.0420	348.06	0.0415	308.13	0.0375	332.13	0.0333	324.17
		18	0.0489	392.00	0.0436	359.65	0.0433	373.44	0.0370	342.18	0.0357	403.59
		21	0.0552	458.51	0.0461	470.82	0.0451	450.61	0.0402	469.34	0.0413	501.58
	(10 , 14)	6	0.0444	224.37	0.0374	242.55	0.0342	249.69	0.0312	251.39	0.0325	254.30
		9	0.0380	290.12	0.0336	295.44	0.0341	287.43	0.0346	276.28	0.0328	282.21
		12	0.0438	319.61	0.0355	286.93	0.0346	296.15	0.0347	278.42	0.0337	308.46
		15	0.0529	314.01	0.0436	334.34	0.0430	341.52	0.0354	322.69	0.0371	329.06
		18	0.0571	379.20	0.0434	360.09	0.0445	365.31	0.0383	380.36	0.0368	394.31
		21	0.0646	410.13	0.0519	414.08	0.0416	484.17	0.0431	466.15	0.0385	436.05
	(10 , 17)	6	0.0483	234.75	0.0333	247.44	0.0401	269.62	0.0343	245.62	0.0333	307.36
		9	0.0363	295.77	0.0367	268.04	0.0364	291.10	0.0374	258.86	0.0294	282.27
		12	0.0351	329.99	0.0411	310.22	0.0407	292.53	0.0357	304.56	0.0349	303.51
		15	0.0509	333.40	0.0448	310.11	0.0395	319.61	0.0368	323.74	0.0351	319.94
		18	0.0570	383.22	0.0486	401.01	0.0458	407.99	0.0384	373.00	0.0418	370.31
		21	0.0662	412.05	0.0522	455.89	0.0524	481.43	0.0453	417.16	0.0460	433.91
	(10 , 20)	6	0.0474	228.54	0.0392	258.81	0.0337	305.49	0.0309	296.10	0.0366	250.68
		9	0.0351	271.66	0.0426	278.14	0.0364	259.08	0.0369	280.78	0.0329	296.92
		12	0.0463	288.03	0.0388	286.33	0.0417	304.29	0.0353	329.39	0.0326	284.35
		15	0.0656	308.46	0.0448	333.51	0.0416	364.54	0.0364	359.71	0.0422	340.87
		18	0.0565	410.95	0.0472	414.96	0.0438	346.47	0.0410	388.27	0.0390	394.91
		21	0.0529	418.09	0.0498	476.86	0.0480	410.35	0.0433	438.80	0.0426	433.25

(40,25,500)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	6	0.0291	295.17	0.0310	286.38	0.0329	266.89	0.0262	234.70	0.0289	245.74
		9	0.0311	287.09	0.0328	286.49	0.0301	288.30	0.0329	268.09	0.0307	278.80
		12	0.0352	302.92	0.0383	297.47	0.0359	278.86	0.0339	300.06	0.0297	308.08
		15	0.0406	371.30	0.0338	373.93	0.0359	339.22	0.0337	356.20	0.0295	335.10
		18	0.0453	401.34	0.0399	367.51	0.0450	373.55	0.0359	413.04	0.0331	404.36
		21	0.0539	414.30	0.0488	408.10	0.0456	373.60	0.0367	456.38	0.0390	502.79
	(10 , 14)	6	0.0396	252.82	0.0325	253.81	0.0284	276.77	0.0322	264.19	0.0317	255.19
		9	0.0383	253.15	0.0358	283.19	0.0323	259.03	0.0294	256.77	0.0309	265.90
		12	0.0457	266.39	0.0331	300.83	0.0316	304.02	0.0306	295.39	0.0308	316.04
		15	0.0413	344.33	0.0408	308.85	0.0358	333.01	0.0333	303.19	0.0343	317.20
		18	0.0503	383.71	0.0404	395.30	0.0400	412.11	0.0366	391.39	0.0339	342.19
		21	0.0495	445.89	0.0447	414.09	0.0389	458.41	0.0396	457.70	0.0393	398.65
	(10 , 17)	6	0.0521	241.01	0.0319	214.10	0.0333	239.97	0.0311	241.56	0.0304	271.66
		9	0.0306	274.08	0.0298	263.92	0.0362	267.99	0.0343	289.73	0.0279	313.57
		12	0.0339	337.85	0.0373	309.18	0.0360	305.88	0.0322	319.61	0.0353	293.19
		15	0.0390	331.70	0.0405	325.33	0.0373	316.43	0.0316	337.96	0.0337	336.20
		18	0.0459	381.13	0.0436	436.11	0.0390	374.48	0.0359	380.85	0.0387	391.07
		21	0.0556	450.78	0.0465	493.45	0.0457	479.39	0.0427	446.71	0.0407	464.83
	(10 , 20)	6	0.0384	232.06	0.0366	256.40	0.0301	273.81	0.0258	252.88	0.0299	246.28
		9	0.0346	248.32	0.0320	305.17	0.0347	288.97	0.0331	286.77	0.0316	276.06
		12	0.0379	268.20	0.0347	295.99	0.0351	278.53	0.0317	324.34	0.0286	321.15
		15	0.0423	343.84	0.0405	358.55	0.0411	302.58	0.0332	339.72	0.0356	376.13
		18	0.0376	408.54	0.0416	450.00	0.0402	407.06	0.0365	415.02	0.0389	450.06
		21	0.0474	414.46	0.0465	460.05	0.0427	495.82	0.0411	485.71	0.0372	403.70

(20,30,600)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	6	0.0751	119.96	0.0410	126.17	0.0404	140.50	0.0373	141.10	0.0391	122.27
		9	0.0544	144.62	0.0428	147.64	0.0453	147.36	0.0444	153.52	0.0475	148.79
		12	0.0904	121.99	0.0491	162.47	0.0411	155.50	0.0442	155.16	0.0515	145.66
		15	0.0640	161.59	0.0601	156.92	0.0490	156.76	0.0636	136.60	0.0562	155.60
		18	0.0722	158.96	0.0613	176.53	0.0565	171.48	0.0466	173.78	0.0511	177.36
		21	0.0738	164.45	0.0622	172.41	0.0598	177.47	0.0546	189.55	0.0606	189.28
		24	0.0756	182.29	0.0721	190.81	0.0683	192.63	0.0577	190.04	0.0609	194.22
	(10 , 14)	6	0.0929	113.36	0.0400	152.31	0.0381	155.88	0.0419	126.17	0.0436	145.17
		9	0.0744	133.36	0.0471	156.65	0.0456	157.47	0.0413	155.39	0.0435	156.15
		12	0.0641	149.34	0.0519	224.70	0.0482	148.73	0.0511	156.48	0.0527	155.77
		15	0.0701	159.33	0.0609	160.99	0.0575	147.86	0.0487	163.67	0.0549	166.86
		18	0.0910	161.65	0.0607	189.11	0.0574	177.74	0.0539	181.59	0.0484	158.18
		21	0.0874	165.77	0.0677	178.40	0.0630	166.48	0.0586	162.58	0.0519	199.93
		24	0.0803	175.60	0.0708	192.90	0.0696	166.97	0.0588	212.84	0.0560	192.57
	(10 , 17)	6	0.0747	122.59	0.0422	143.03	0.0442	136.88	0.0368	138.19	0.0475	147.86
		9	0.0534	139.90	0.0472	151.65	0.0513	150.00	0.0452	160.54	0.0532	140.28
		12	0.0527	148.96	0.0519	177.95	0.0499	178.12	0.0557	130.29	0.0418	167.03
		15	0.0627	178.73	0.0594	160.38	0.0594	158.73	0.0585	212.45	0.0535	168.84
		18	0.0820	236.68	0.0691	186.52	0.0615	186.91	0.0600	194.22	0.0555	180.70
		21	0.0767	167.96	0.0616	180.10	0.0640	156.65	0.0562	176.04	0.0569	165.87
		24	0.0834	172.25	0.0672	189.99	0.0678	177.90	0.0655	187.35	0.0624	219.54
	(10 , 20)	6	0.0915	104.80	0.0658	124.90	0.0515	135.33	0.0460	142.92	0.0404	142.26
		9	0.0506	138.90	0.0514	132.70	0.0456	162.30	0.0439	153.30	0.0476	155.44
		12	0.0895	138.69	0.0520	163.30	0.0545	172.02	0.0503	154.78	0.0475	173.78
		15	0.0721	155.22	0.0506	179.11	0.0552	169.06	0.0574	183.12	0.0475	178.67
		18	0.0670	161.65	0.0593	169.45	0.0629	195.92	0.0515	173.84	0.0534	182.95
		21	0.0691	194.00	0.0626	189.38	0.0561	231.68	0.0548	258.25	0.0579	227.83
		24	0.0632	202.84	0.0702	197.95	0.0636	209.49	0.0650	217.23	0.0606	199.49

(20,30,600)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	6	0.0416	104.16	0.0454	121.99	0.0370	128.69	0.0396	132.42	0.0333	133.69
		9	0.0726	108.39	0.0396	134.95	0.0372	130.17	0.0351	133.25	0.0367	128.14
		12	0.0537	128.16	0.0400	147.59	0.0432	137.92	0.0349	152.36	0.0394	131.83
		15	0.0623	155.13	0.0524	124.68	0.0501	143.35	0.0412	190.98	0.0408	173.67
		18	0.0583	158.13	0.0570	144.17	0.0539	121.50	0.0419	185.15	0.0396	176.31
		21	0.0883	140.14	0.0607	152.25	0.0486	180.43	0.0446	190.15	0.0465	195.70
		24	0.0752	137.04	0.0665	169.01	0.0635	169.01	0.0491	184.77	0.0497	209.37
	(10 , 14)	6	0.0455	122.86	0.0436	118.97	0.0353	151.54	0.0336	155.39	0.0355	120.29
		9	0.0591	150.33	0.0496	168.24	0.0393	137.09	0.0329	143.30	0.0356	135.11
		12	0.0632	133.69	0.0457	130.11	0.0457	130.89	0.0431	145.23	0.0419	131.65
		15	0.0587	120.42	0.0601	130.28	0.0494	130.94	0.0449	178.18	0.0410	157.36
		18	0.0664	159.67	0.0547	156.93	0.0452	170.32	0.0440	179.06	0.0449	169.61
		21	0.0780	187.17	0.0547	169.50	0.0515	170.71	0.0498	207.79	0.0402	177.19
		24	0.0960	186.11	0.0656	164.77	0.0661	160.27	0.0488	203.28	0.0483	183.12
	(10 , 17)	6	0.0896	119.95	0.0372	136.22	0.0372	124.90	0.0310	134.07	0.0381	148.03
		9	0.0848	125.01	0.0476	150.06	0.0485	134.90	0.0435	153.30	0.0360	145.77
		12	0.0703	126.88	0.0396	155.00	0.0398	163.78	0.0397	167.14	0.0419	171.43
		15	0.0598	156.87	0.0416	148.79	0.0438	182.79	0.0399	241.46	0.0423	147.91
		18	0.0694	156.98	0.0610	152.48	0.0555	190.87	0.0452	185.54	0.0449	183.56
		21	0.0840	147.91	0.0578	189.99	0.0566	158.68	0.0528	186.80	0.0471	205.25
		24	0.0664	191.64	0.0652	186.42	0.0557	196.80	0.0565	187.85	0.0508	195.42
	(10 , 20)	6	0.0766	98.21	0.0438	143.41	0.0396	156.75	0.0376	132.54	0.0427	112.66
		9	0.0547	138.03	0.0445	135.01	0.0353	165.82	0.0365	166.76	0.0369	141.15
		12	0.0616	115.73	0.0393	154.12	0.0423	150.77	0.0389	160.33	0.0357	145.88
		15	0.0484	175.99	0.0498	177.74	0.0410	153.35	0.0413	166.26	0.0439	186.53
		18	0.0473	157.42	0.0660	145.94	0.0582	156.54	0.0437	170.38	0.0448	187.35
		21	0.0838	191.69	0.0530	189.43	0.0562	281.99	0.0483	190.21	0.0471	192.02
		24	0.0609	172.52	0.0493	183.89	0.0560	205.36	0.0513	207.40	0.0515	185.32

(20,30,600)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	6	0.0633	102.06	0.0480	97.55	0.0332	110.79	0.0364	124.80	0.0337	148.24
		9	0.0665	123.85	0.0645	105.13	0.0319	120.56	0.0383	143.96	0.0308	105.57
		12	0.0801	111.23	0.0832	119.41	0.0451	119.57	0.0310	125.45	0.0366	149.73
		15	0.0632	137.53	0.0811	129.02	0.0541	117.26	0.0374	180.38	0.0345	168.02
		18	0.0866	120.29	0.0582	116.28	0.0535	130.11	0.0409	168.73	0.0346	173.95
		21	0.0768	142.26	0.0603	149.34	0.0459	172.14	0.0399	203.34	0.0430	185.86
		24	0.0637	146.44	0.0774	141.43	0.0571	156.10	0.0449	151.65	0.0439	182.19
	(10 , 14)	6	0.0844	107.05	0.0806	114.41	0.0360	114.96	0.0312	170.32	0.0312	161.26
		9	0.0693	129.13	0.0368	198.61	0.0362	142.20	0.0352	120.67	0.0341	151.04
		12	0.0728	125.07	0.0402	147.91	0.0367	151.87	0.0373	151.43	0.0371	149.62
		15	0.0813	120.78	0.0715	116.17	0.0422	123.91	0.0332	123.69	0.0364	184.93
		18	0.0833	123.20	0.0485	171.20	0.0418	176.81	0.0419	191.53	0.0326	167.85
		21	0.0610	164.45	0.0552	137.97	0.0482	155.38	0.0444	213.34	0.0343	180.43
		24	0.0933	146.10	0.0516	178.51	0.0461	219.48	0.0444	180.87	0.0407	172.91
	(10 , 17)	6	0.0715	107.60	0.0308	158.13	0.0389	117.38	0.0296	145.94	0.0327	141.54
		9	0.0298	164.34	0.0572	110.62	0.0513	104.63	0.0327	144.13	0.0327	151.87
		12	0.0354	145.88	0.0305	162.64	0.0378	171.54	0.0397	168.45	0.0361	177.96
		15	0.0389	167.63	0.0448	181.42	0.0444	196.85	0.0373	229.10	0.0362	189.33
		18	0.0752	140.06	0.0506	163.18	0.0442	203.78	0.0418	164.72	0.0390	186.80
		21	0.0640	157.14	0.0449	186.04	0.0516	188.99	0.0402	192.07	0.0424	178.51
		24	0.0786	162.96	0.0505	165.10	0.0582	178.07	0.0448	199.27	0.0472	207.62
	(10 , 20)	6	0.0351	105.29	0.0338	112.98	0.0337	150.01	0.0345	147.42	0.0343	129.24
		9	0.0364	132.37	0.0347	148.02	0.0350	163.96	0.0271	154.23	0.0338	142.69
		12	0.0640	112.43	0.0396	158.40	0.0368	168.24	0.0304	173.40	0.0358	166.58
		15	0.0700	147.75	0.0364	176.26	0.0382	178.45	0.0361	175.98	0.0389	188.40
		18	0.0304	173.01	0.0712	133.42	0.0400	155.71	0.0361	179.28	0.0425	177.46
		21	0.0448	194.66	0.0474	254.85	0.0389	247.82	0.0475	203.45	0.0419	211.24
		24	0.0512	175.71	0.0413	208.06	0.0442	211.29	0.0458	220.03	0.0427	214.86

(40,30,600)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.75)	(10 , 11)	6	0.0336	344.38	0.0273	354.49	0.0273	339.71	0.0259	337.57	0.0226	364.92
		9	0.0294	389.53	0.0255	392.27	0.0270	396.84	0.0291	352.51	0.0249	353.07
		12	0.0368	350.04	0.0279	384.97	0.0317	413.04	0.0301	411.55	0.0267	414.42
		15	0.0565	384.58	0.0348	401.95	0.0342	402.76	0.0321	415.73	0.0273	459.68
		18	0.0499	481.26	0.0366	531.90	0.0385	430.40	0.0318	454.51	0.0307	460.50
		21	0.0522	560.02	0.0402	562.87	0.0420	468.52	0.0347	512.84	0.0330	561.23
		24	0.0695	539.25	0.0484	570.56	0.0458	550.19	0.0404	531.79	0.0351	539.75
	(10 , 14)	6	0.0319	351.97	0.0237	368.66	0.0271	371.24	0.0260	351.25	0.0250	351.30
		9	0.0356	343.95	0.0316	390.46	0.0311	348.28	0.0266	385.91	0.0259	390.30
		12	0.0366	397.61	0.0328	413.97	0.0293	405.68	0.0303	400.24	0.0288	435.61
		15	0.0478	416.11	0.0377	464.78	0.0349	425.57	0.0318	462.86	0.0284	424.57
		18	0.0517	489.27	0.0427	503.23	0.0341	461.81	0.0353	511.25	0.0303	506.63
		21	0.0580	426.99	0.0469	561.06	0.0435	530.30	0.0357	494.66	0.0324	543.82
		24	0.0517	632.47	0.0481	565.84	0.0458	548.92	0.0423	577.32	0.0405	650.15
	(10 , 17)	6	0.0497	310.17	0.0295	360.31	0.0251	358.05	0.0219	388.98	0.0248	355.70
		9	0.0339	402.43	0.0356	387.12	0.0302	409.36	0.0279	357.24	0.0257	339.93
		12	0.0336	397.99	0.0324	491.96	0.0347	446.33	0.0292	386.18	0.0252	402.38
		15	0.0369	439.90	0.0420	493.29	0.0372	425.62	0.0329	442.10	0.0311	433.53
		18	0.0484	479.17	0.0500	474.17	0.0371	538.88	0.0365	467.36	0.0306	504.82
		21	0.0484	568.81	0.0445	577.21	0.0423	510.70	0.0343	543.43	0.0398	532.56
		24	0.0521	699.92	0.0504	611.10	0.0475	608.63	0.0395	574.41	0.0376	681.68
	(10 , 20)	6	0.0256	319.89	0.0310	343.12	0.0282	338.17	0.0308	378.21	0.0244	350.54
		9	0.0352	367.39	0.0341	368.66	0.0281	453.68	0.0266	377.50	0.0265	336.47
		12	0.0434	469.01	0.0329	436.22	0.0341	402.83	0.0313	424.52	0.0284	435.56
		15	0.0404	434.24	0.0362	464.18	0.0329	478.46	0.0348	502.07	0.0296	505.92
		18	0.0372	476.70	0.0482	506.74	0.0388	441.88	0.0330	524.92	0.0354	535.03
		21	0.0444	553.81	0.0400	567.99	0.0392	609.67	0.0382	558.75	0.0320	553.37
		24	0.0384	630.49	0.0437	748.74	0.0382	559.69	0.0417	625.00	0.0388	599.40

(40,30,600)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.85)	(10 , 11)	6	0.0490	251.57	0.0236	390.03	0.0258	342.02	0.0234	391.73	0.0212	334.44
		9	0.0378	318.62	0.0239	331.69	0.0272	357.51	0.0258	365.59	0.0221	367.67
		12	0.0561	324.99	0.0279	420.56	0.0283	416.22	0.0271	405.41	0.0267	390.36
		15	0.0428	482.80	0.0386	430.89	0.0292	481.25	0.0328	416.94	0.0272	416.72
		18	0.0734	453.14	0.0385	472.80	0.0383	470.06	0.0328	415.07	0.0319	443.90
		21	0.0486	534.10	0.0432	530.03	0.0380	509.32	0.0315	523.11	0.0354	584.73
		24	0.0666	470.11	0.0466	635.38	0.0446	548.60	0.0371	589.79	0.0355	623.02
	(10 , 14)	6	0.0264	399.91	0.0213	372.23	0.0232	365.37	0.0266	358.66	0.0237	395.80
		9	0.0336	355.09	0.0314	393.81	0.0284	360.37	0.0266	353.67	0.0225	433.31
		12	0.0391	429.57	0.0317	413.04	0.0328	395.74	0.0282	390.24	0.0275	441.87
		15	0.0411	494.60	0.0395	412.54	0.0301	529.53	0.0311	441.11	0.0286	458.52
		18	0.0492	460.50	0.0415	544.10	0.0375	559.69	0.0325	525.63	0.0295	521.68
		21	0.0464	472.58	0.0430	572.60	0.0411	504.27	0.0328	557.77	0.0323	581.50
		24	0.0608	519.98	0.0446	622.47	0.0451	599.56	0.0395	582.76	0.0381	541.23
	(10 , 17)	6	0.0336	346.42	0.0278	338.01	0.0256	381.35	0.0242	412.60	0.0248	366.68
		9	0.0301	393.66	0.0325	436.77	0.0285	406.94	0.0260	427.93	0.0260	398.37
		12	0.0334	519.43	0.0324	391.35	0.0330	468.02	0.0285	423.97	0.0236	444.85
		15	0.0323	509.38	0.0363	455.49	0.0381	424.08	0.0307	515.09	0.0312	427.81
		18	0.0486	456.71	0.0500	477.80	0.0378	493.06	0.0357	551.78	0.0312	422.71
		21	0.0458	572.16	0.0354	520.47	0.0394	490.98	0.0331	595.28	0.0370	494.33
		24	0.0452	613.03	0.0431	675.52	0.0438	592.75	0.0415	573.20	0.0356	625.66
	(10 , 20)	6	0.0204	314.94	0.0264	414.90	0.0252	395.02	0.0283	355.70	0.0241	406.50
		9	0.0290	339.88	0.0328	335.82	0.0237	457.64	0.0236	418.59	0.0223	399.53
		12	0.0416	373.33	0.0360	444.13	0.0304	457.53	0.0306	414.14	0.0280	457.64
		15	0.0366	469.22	0.0343	503.83	0.0329	488.72	0.0339	490.26	0.0259	448.19
		18	0.0287	485.54	0.0473	460.99	0.0356	549.09	0.0305	496.59	0.0340	550.80
		21	0.0413	543.87	0.0366	537.00	0.0367	519.60	0.0354	550.91	0.0335	569.19
		24	0.0432	680.97	0.0402	588.19	0.0349	676.90	0.0399	723.32	0.0377	611.54

(40,30,600)	Procurement Cost	Number of Primary Machine Groups	Operating Cost									
			(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization			GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time	GAP	CPU Time
(0.65 , 0.95)	(10 , 11)	6	0.0301	223.33	0.0194	353.94	0.0199	248.98	0.0192	319.34	0.0179	370.36
		9	0.0333	267.10	0.0257	248.54	0.0200	377.94	0.0214	407.43	0.0186	340.48
		12	0.0355	306.65	0.0281	304.40	0.0240	401.72	0.0218	376.51	0.0215	383.99
		15	0.0432	315.88	0.0375	328.45	0.0280	353.39	0.0262	465.77	0.0234	495.76
		18	0.0452	372.34	0.0298	426.77	0.0319	391.72	0.0283	419.19	0.0275	382.17
		21	0.0471	406.50	0.0336	394.86	0.0333	561.67	0.0270	622.58	0.0298	600.33
		24	0.0508	406.33	0.0387	526.90	0.0376	486.14	0.0334	495.64	0.0305	617.37
	(10 , 14)	6	0.0215	326.92	0.0214	290.67	0.0197	230.08	0.0191	399.09	0.0201	423.97
		9	0.0336	276.22	0.0278	367.73	0.0206	342.84	0.0192	439.24	0.0195	412.93
		12	0.0317	321.42	0.0250	459.01	0.0242	362.78	0.0236	439.51	0.0222	419.03
		15	0.0329	466.81	0.0328	432.48	0.0283	378.00	0.0248	473.79	0.0207	468.24
		18	0.0336	531.46	0.0343	548.82	0.0317	487.03	0.0253	490.98	0.0233	515.09
		21	0.0341	679.92	0.0303	538.54	0.0333	638.40	0.0314	514.71	0.0268	527.51
		24	0.0548	483.89	0.0399	543.05	0.0356	576.23	0.0345	553.65	0.0286	553.60
	(10 , 17)	6	0.0202	345.65	0.0213	395.52	0.0200	331.80	0.0178	371.19	0.0193	402.77
		9	0.0243	422.65	0.0257	302.53	0.0255	354.05	0.0226	407.11	0.0183	388.76
		12	0.0310	443.52	0.0266	455.67	0.0243	430.40	0.0229	498.34	0.0186	429.90
		15	0.0251	514.98	0.0286	480.38	0.0265	447.86	0.0262	499.77	0.0265	489.99
		18	0.0391	477.57	0.0364	497.24	0.0284	530.42	0.0281	403.70	0.0227	514.38
		21	0.0405	624.45	0.0310	536.68	0.0307	464.62	0.0281	520.42	0.0297	522.89
		24	0.0378	671.79	0.0370	657.29	0.0380	749.95	0.0330	530.69	0.0319	725.01
	(10 , 20)	6	0.0220	324.22	0.0190	411.94	0.0176	432.37	0.0256	438.97	0.0187	365.14
		9	0.0237	368.99	0.0287	425.84	0.0200	404.37	0.0201	404.31	0.0211	415.96
		12	0.0326	371.85	0.0287	504.93	0.0223	505.75	0.0230	356.13	0.0204	431.38
		15	0.0305	444.73	0.0291	514.54	0.0247	438.91	0.0268	497.79	0.0206	477.91
		18	0.0272	454.12	0.0338	474.01	0.0329	508.11	0.0242	554.15	0.0290	565.79
		21	0.0288	587.87	0.0360	649.22	0.0294	605.06	0.0259	622.36	0.0289	564.09
		24	0.0408	673.71	0.0340	659.49	0.0266	615.33	0.0312	662.89	0.0273	681.08

Means		Operating Cost									
		(0.1 , 0.3)		(0.1 , 0.7)		(0.1 , 1.1)		(0.1 , 1.5)		(0.1 , 1.9)	
Utilization	**Procurement Cost**	Mean GAP	Mean CPU Time	Mean GAP	Mean CPU Time	Mean GAP	Mean CPU Time	Mean GAP	Mean CPU Time	Mean GAP	Mean CPU Time
(0.65 , 0.75)	(10 , 11)	0.0565	127.16	0.0497	124.85	0.0426	127.21	0.0395	136.30	0.0378	138.52
	(10 , 14)	0.0557	161.33	0.0450	170.67	0.0398	171.30	0.0371	175.91	0.0359	177.01
	(10 , 17)	0.0529	174.62	0.0423	176.82	0.0408	176.64	0.0383	177.74	0.0357	183.05
	(10 , 20)	0.0468	169.66	0.0432	184.97	0.0405	185.59	0.0382	186.64	0.0368	185.99
(0.65 , 0.85)	(10 , 11)	0.0598	162.77	0.0497	170.41	0.0460	170.00	0.0427	175.76	0.0414	176.09
	(10 , 14)	0.0605	169.80	0.0504	172.12	0.0465	175.40	0.0426	177.00	0.0412	180.05
	(10 , 17)	0.0641	172.97	0.0506	175.89	0.0477	178.06	0.0445	181.65	0.0420	178.67
	(10 , 20)	0.0548	168.53	0.0506	174.10	0.0456	181.79	0.0439	183.54	0.0419	178.75
(0.65 , 0.95)	(10 , 11)	0.0546	151.44	0.0473	154.33	0.0408	160.51	0.0378	173.41	0.0361	176.81
	(10 , 14)	0.0557	161.33	0.0450	170.67	0.0398	171.30	0.0371	175.91	0.0359	177.01
	(10 , 17)	0.0529	174.62	0.0423	176.82	0.0408	176.64	0.0383	177.74	0.0357	183.05
	(10 , 20)	0.0468	169.66	0.0432	184.97	0.0405	185.59	0.0382	186.64	0.0368	185.99

REFERENCES

Achugbue, J.O. and F.Y. Chin, "Scheduling the Open Shop to Minimize Mean Flow Time," *SIAM Journal on Computing*, Vol. 9 (1982), 306-311.

Adiri, I. and N. Aizikowitz (Hefetz), "Open-shop Scheduling Problems with Dominated Machines," *Naval Research Logistics*, Vol. 36 (1989), 273-281.

Adiri, I. and N. Amit, "Openshop and Flowshop Scheduling to Minimize Sum of Completion Times," *Computers and Operations Research*, Vol. 11 (1984), 275-284.

Ahmadi J., S. Grotzinger, and D. Johnson, "Component Allocation and Partitioning for a Dual Delivery Placement Machine," *Operations Research*, Vol. 36, No. 2 (1988), 176-191.

Amini, M.M. and M. Racer, "A Rigorous Computational Comparison of Alternative Solution Methods for the Generalized Assignment Problem," *Management Science*, Vol. 40, No. 7 (1994), 868-889.

Ammons, J.C., C.B. Lofgren, and L.F. McGinnis, "A Large Scale Machine Loading Problem in Flexible Assembly," *Annals of Operations Research*, Vol. 3 (1985), 319-332.

Angel, D.P., *Restructuring for Innovation: The Remaking of the U.S. Semiconductor Industry*, Guilford Press, New York, NY (1995).

Askin, R.G., M. Dror, and A.J. Vakharia, "Printed Circuit Board Family Grouping and Component Allocation, Open-Shop Assembly Cell," *Naval Research Logistics*, Vol. 41 (1994), 587-608.

Askin, R.G., H.M. Selim, and A.J. Vakharia, "A Methodology for Designing Flexible Cellular Manufacturing Systems," *IIE Transactions*, Vol. 29 (1997), 599-610.

Balachandran, V., "An Integer Generalized Transportation Model for Optimal Job Assignment in Computer Networks," Working Paper 34-72-3 (1972), Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA.

Balakrishnan, A. and F. Vanderbeck, "A Tactical Planning Model for Mixed-Model Electronics Assembly Operations," *Operations Research*, Vol. 47, No. 3 (1999), to appear.

Ball, M.O. and M.J. Magazine, "Sequencing of Insertions in Printed Circuit Board Assemblies," *Operations Research*, Vol. 36, No. 2 (1988), 192-201.

Barcia, P. and K. Jörnsten, "Improved Lagrangean Decomposition: An Application to the Generalized Assignment Problem," *European Journal of Operational Research*, Vol. 49 (1990), 84-92

Ben-Arieh, D. and M. Dror, "Part Assignment to Electronic Insertion Machines: Two Machine Case," *International Journal of Production Research*, Vol. 28, No. 7 (1990), 1317-1327.

Bitran, G.R., Hass, E.A., and A.C. Hax, "Hierarchical Production Planning: A Single Stage System," *Operations Research*, Vol. 29, No. 4 (1981), 717-743.

Bitran, G.R., Hass, E.A., and A.C. Hax, "Hierarchical Production Planning: A Two Stage System," *Operations Research*, Vol. 30, No. 2 (1982), 232-251.

Carmon T.F., O.Z. Maimon, and E.M. Dar-El, "Group Set-up for Printed Circuit Board Assembly," *International Journal of Production Research*, Vol. 27, No. 10 (1989), 1795-1810.

Cattrysse, D.G. and L.N. Van Wassenhove, "A Survey of Algorithms for the Generalized Assignment Problem," *European Journal of Operational Research,* Vol. 60 (1992), 260-272.

Cornuejols, G., M.L. Fisher, and G.L. Nemhauser, "Location of Bank Accounts to Optimize Float: An Analytic Study of Exact and Approximate Algorithms," *Management Science*, Vol. 23 (1977), 789-810.

Crama Y., A.W.J. Kolen, A.G. Oerlemans, and F.C.R. Spieksma, "Throughput Rate Optimization in the Automated Assembly of Printed Circuit Boards," *Annals of Operations Research*, Vol. 26 (1990), 455-480.

De Witte, J., "The Use of Similarity Coefficients in Production Flow Analysis," *International Journal of Production Research*, Vol. 18, No. 4 (1980), 503-514.

Demeester, L. and C.S. Tang, "Reducing Cycle Time at an IBM Wafer Fabrication Facility," *Interfaces,* Vol. 26, No. 2 (1996), 34-49.

Drezner, Z. and S. Nof, "On Optimizing Bin Packing and Insertion Plans for Assembly Robots," *IIE Transactions*, Vol. 16, No. 3 (1984), 262-270.

Dror, M., "Openshop Scheduling with Machine Dependent Processing Times," *Discrete Applied Mathematics*, Vol. 39 (1992), 197-205

Erlenkotter, D., "A Dual-Based Procedure for Uncapacitated Facility Location," *Operations Research*, Vol. 26, No. 6 (1978), 992-1009.

Fathi, Y. and J. Taheri, "A Mathematical Model for Loading the Sequencers in a Printed Circuit Pack Manufacturing Environment," *International Journal of Production Research*, Vol. 27 (1989), 1305-1316.

Fisher, M.L., "The Lagrangian Relaxation Method for Solving Integer Programming Problems," *Management Science,* Vol. 27, No. 1 (1981), 1-18.

Fisher, M.L., "An Application Oriented Guide to Lagrangian Relaxation," *Interfaces*, Vol. 15, No. 2 (1985), 10-21.

Fisher, M.L. and R. Jaikumar, "A Generalized Assignment Heuristic for Vehicle Routing," *Networks*, Vol. 11 (1981), 109-124

Fisher, M.L., R. Jaikumar, and L.N. Van Wassenhove, "A Multiplier Adjustment Method for the Generalized Assignment Problem," *Management Science*, Vol. 32, No. 9 (1986), 1095-1103.

Francis, R.L., H.W. Hamacher, C-Y. Lee, and S. Yeralan, "On Automating Robotic Assembly Workplace Planning," *Research Report* (1989), Industrial and Systems Engineering Department, University of Florida, Gainesville, FL.

Garey, M.R. and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, San Francisco, CA (1979).

Gavish, B. and H. Pirkul, "Computer and Database Location in Distributed Computer Systems," *IEEE Transactions Computers*, Vol. 35, No. 7 (1986), 583-590.

Gavish, B. and H. Pirkul, "Algorithms for the Multi-Resource Generalized Assignment Problem," *Management Science*, Vol. 37, No. 6 (1991), 695-713.

Geoffrion, A.M., "Lagrangean Relaxation for Integer Programming," *Mathematical Programming Study*, Vol. 2 (1974), 82-114.

Geoffrion, A.M. and R. McBride, "Lagrangean Relaxation Applied to Capacitated Facility Location Problems," *IIE Transactions*, Vol. 10, No. 1 (1978), 40-47.

Golovin, J.J., "A Total Framework for Semiconductor Production Planning and Scheduling", *Solid State Technology* (May 1986), 160-170.

Gonzalez, T. and S. Sahni, "Open Shop Scheduling to Minimize Finish Time," *Journal of the Association for Computing Machinery*, Vol. 23, No. 4 (1976), 665-679.

Gonzalez, T., "Unit Execution Time Shop Problems," *Mathematics of Operations Research*, Vol. 7 (1982), 57-66.

Gray, P., "Exact Solution of the Fixed-Charge Transportation Problem," *Operations Research*, Vol. 19 (1971), 1259-1538.

Griogoriadis, M.D., D.J. Tang, and L.S. Woo, "Considerations in the Optimal Synthesis of Some Communications Networks," presented at the 45th Joint National Meeting of ORSA/TIMS (1974), Boston, MA.

Gross, D. and C.E. Pinkus, "Optimal Allocation of Ships to Yards for Regular Overhauls," *Technical Memorandum 63095* (1972), Institute of Management Science Engineering, George Washington University, Washington, DC.

Grotzinger, S., "Feeder Assignment Models for Concurrent Placement Machines," *IIE Transactions*, Vol. 24, No. 4 (1992) 31-46.

Guignard, M. and S. Kim, "Lagrangean Decomposition: A Model Yielding Stronger Lagrangean Bounds," *Mathematical Programming*, Vol. 39 (1987), 215-228.

Guignard, M. and M. Rosenwein, "An Improved Dual-Based Algorithm for the Generalized Assignment Problem," *Operations Research*, Vol. 37, No. 4 (1989), 658-663.

Hung, Y.F. and Q.Z. Wang, "A New Formulation Technique for Alternative Material Planning-An Approach for Semiconductor Bin Allocation Planning," *Computers & Industrial Engineering*, Vol. 32, No. 2 (1997), 281-297.

Jörnsten, K. and M. Näsberg, "A New Lagrangian Relaxation Approach to the Generalized Assignment Problem," *European Journal of Operational Research*, Vol. 27 (1986), 313-323.

Kear, F.W., *Printed Circuit Assembly Manufacturing*, Marcel Dekker, Inc., New York, NY (1987).

Kellerer, H., T. Tautenhahn, and G. Woeginger, "Note: Open-Shop Scheduling with Release Dates to Minimize Maximum Lateness," *Naval Research Logistics*, Vol. 42 (1995), 141-145.

King, R.E., J.A. Jones, and C.T. Culbreth, "A Comprehensive Review of Manufacturing Cell Design," *Working Paper* (1994), Furniture Manufacturing and Management Center, North Carolina State University, Raleigh, NC.

Klastorin, T.D., "The p-Median Problem for Cluster Analysis: A Comparative Test Using the Mixture Model Approach," *Management Science*, Vol. 31, No. 1 (1985), 84-95.

Lawler, E.L., J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys, "Sequencing and Scheduling: Algorithms and Complexity," *Report 8945/A* (1989), Erasmus University, Rotterdam, The Netherlands.

Leachman, R.C., "Modeling Techniques for Automated Production Planning in the Semiconductor Industry," *Optimization in Industry*, ed. T.A. Ciriani and R.C. Leachman, John Wiley &Sons Ltd. (1993).

Leachman, R.C., Benson, R.F., Liu, C., and D.J. Raar, "IMPReSS: An Automated Production Planning and Delivery-Quotation System at Harris Corporation - Semiconductor Sector," *Interfaces*, Vol. 26, No. 1 (1996), 6-37.

Leachman, R.C. and T.F. Carmon, "On Capacity Modeling for Production Planning with Alternative Machine Types," *IIE Transactions*, Vol. 24, No. 4 (1992), 62-72.

Lee, Y., Kim, S., Yea, S., and B. Kim, "Production Planning in Semiconductor Wafer Fab Considering Variable Cycle Times," *Computers & Industrial Engineering*, Vol. 33, Nos. 3-4 (1997), 713-716.

Leon, V.J. and B.A. Peters, "Replanning and Analysis of Partial Setup Strategies in Printed Circuit Board Assembly Systems," *International Journal of Flexible Manufacturing Systems*, Vol. 8, No. 4 (1996), 389-412

Lofgren, C.B. and L.F. McGinnis, "Soft Configuration in Automated Insertion," Proceedings of the 1986 IEEE Conference on Robotics and Automation, San Francisco, CA (1986), 138-142.

Lofgren, C.B., L.F. McGinnis and C.A. Tovey, "Routing Printed Circuit Cards Through an Assembly Cell," *Operations Research*, Vol. 39, No. 6 (1991), 992-1104.

Maimon, O. and A. Shtub, "Grouping Methods for Printed Circuit Board Assembly," *International Journal of Production Research*, Vol. 29, No. 7 (1991), 1379-1390.

Malerbo, F., *The Semiconductor Business*, The University of Wisconsin Press, Madison, WI (1985).

Martello, S. and P. Toth, "An Algorithm for the Generalized Assignment Problem," *Operational Research '81*, J.P. Brans (ed.), North-Holland, Amsterdam (1981), 589-603.

Martello, S. and P. Toth, "Linear Assignment Problems," *Annals of Discrete Mathematics*, Vol. 31 (1987), 259-282.

Mazzola, J.B. and A.W. Neebe, "Resource-Constrained Assignment Scheduling," *Operations Research*, Vol. 34, No. 4 (1986), 560-572.

McGinnis, L.F., J.C. Ammons, M. Carlyle, L. Cranmer, G.W. DePuy, K.P. Ellis, C.A. Tovey, and H. Xu, "Automated Process Planning for Printed Circuit Card Assembly," *IIE Transactions*, Vol. 24, No. 4 (1992), 18-30.

McIvor, R., *Managing for Profit in the Semiconductor Industry*, Prenctice Hall, Englewood Cliffs, NJ (1989).

Mulvey, J.M. and M.P. Beck, "Solving Capacitated Clustering Problems," *European Journal of Operational Research*, Vol. 18 (1984), 339-348.

Mulvey, J.M. and H.P. Crowder, "Cluster Analysis: An Application of Lagrangian Relaxation," *Management Science*, Vol. 25, No. 4 (1979), 329-340.

Murphy, R.A., "A Private Fleet Model with Multi-Stop Backhaul," *Working Paper 103* (1986), Optimal Decision Systems, Green Bay, WI.

Murty, K.G., "Solving the Fixed Charge Problem by Ranking the Extreme Points," *Operations Research*, Vol. 16 (1967), 268-279.

Nemhauser, G.L. and L.A. Wolsey, *Integer and Combinatorial Optimization*, John Wiley & Sons, Inc., New York, NY (1988).

Padillo, J.M. and D. Meyersdorf, "*A Strategic Domain: IE in the Semiconductor Industry*," *IIE Solutions* Vol. 30, No. 3 (1998), 36-4.

Papadimitriou, C.H. and K. Steiglitz,, *Combinatorial Optimization: Algorithms and Complexity*, Prentice-Hall, Inc., Englewood Cliffs, NJ (1982).

Parker, R.G. and R.L. Rardin, *Discrete Optimization*, Academic Press, Inc. (1988).

Peters, B.A. and G.S. Subramanian, "Analysis of Partial Setup Strategies for Solving the Operational Planning Problem in Parallel Machine Electronic Assembly Systems," *International Journal of Production Research*, Vol. 34, No. 4 (1996), 999-1021.

Pletsch, B., *Integrated Circuits: Making the Miracle Chip*, Pletsch & Associates Austin, TX (1985).

Rajan, A. and M. Segal, "Assigning Components to Robotic Workcells for Electronic Assembly," *AT&T Technical Journal* (May/June 1989), 93-102.

Rao, M.R., "Cluster Analysis and Mathematical Programming," *Journal of the American Statistical Association*, Vol. 66, No. 335 (1971) 622-626.

Ross, G.T. and R.M. Soland, "A Branch and Bound Algorithm for the Generalized Assignment Problem," *Mathematical Programming,* Vol. 8 (1975), 92-103.

Ross, G.T. and R.M. Soland, "Modeling Facility Location Problems as Generalized Assignment Problems," *Management Science*, Vol. 24, No. 3 (1977), 345-357.

Salkin, H.M. and K. Mathur, *Foundations of Integer Programming*, Elsevier Science Publishing Co., Inc., New York, NY (1989).

Schaller, J.E., S.S. Erenguc, and A.J. Vakharia, "A Methodology for Integrating Cell Formation and Production Planning in Cellular Manufacturing," *Annals of Operations Research*, in press.

Selim, H.M., "A Flexible Cell Formation Approach for Cellular Manufacturing," Ph.D. Dissertation (1993), College of Business and Public Administration, University of Arizona, Tucson, AZ.

Selim, H.M., R.G. Askin, and A.J. Vakharia, "Cell Formation in Group Technology: Review, Evaluation and Directions for Future Research," *Computers & Industrial Engineering*, Vol. 34, No. 1 (1998), 3-20.

Sullivan, G. and K. Fordyce, "IBM Burlington's Logistics Management System," *Intefaces*, Vol. 20, No. 1 (1990), 43-64.

Tang, C.S., "A Max-Min Allocation Problem: Its Solutions and Applications," *Operations Research*, Vol. 36, No. 2 (1988), 359-367.

Technical Forecast: 1997, Version 7, Price Waterhouse World Technology Center, Menlo Park, CA (January 1997).

Technical Forecast: 1999, Price Waterhouse World Technology Center, Menlo Park, CA (October 1998).

Trick, M.A., "A Linear Relaxation Heuristic for the Generalized Assignment Problem," *Naval Research Logistic,* Vol. 39 (1992), 137-151.

Uzsoy, R., Lee, C.Y., and L.A. Martin-Vega, "A Review of Production Planning and Scheduling Models in the Semiconductor Industry. Part I: System Characteristics, Performance Evaluation and Production Planning," *IIE Transactions*, Vol. 24, No. 4 (1992), 47-60.

Vairaktarakis, G. and S. Sahni, "Dual Criteria Preemptive Open-Shop Problems with Minimum Makespan," *Naval Research Logistics*, Vol. 42 (1995), 103-121.

Vakharia, A.J. and B. Çatay, "Two Machine Openshop Scheduling with Machine-Dependent Processing Times," *Discrete Applied Mathematics*, Vol. 73 (1997), 283-288.

Vakharia, A.J. and U. Wemmerlöv, "A Comparative Investigation of Hierarchical Clustering Techniques and Dissimilarity Measures Applied to the Cell Formation Problem," *Journal of Operations Management*, Vol. 13, No. 2 (1995), 117-138.

Wilhelm W.E. and J. Fowler, "Research Directions in Electronics Manufacturing," *IIE Transactions*, Vol. 24, No. 4 (1992), 6-17.

BIOGRAPHICAL SKETCH

Bülent Çatay was born in Istanbul, Turkey. He completed Saint Joseph French High School in 1988 and received his Bachelor of Science degree in industrial engineering from Istanbul Technical University in 1992. He is a member of Alpha Iota Delta Decision Sciences Honorary Society, Institute for Operations Research and Management Science, Institute of Industrial Engineers, Production and Operations Management Society, and American Production and Inventory Control Society.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

S. Selcuk Erenguc, Chairman
Professor of Decision and Information
Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Asoo J. Vakharia, Co-chair
Associate Professor of Decision and
Information Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Daniel J. Conway

Daniel J. Conway
Assistant Professor of Decision and
Information Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Sherman Bai

Sherman X. Bai
Assistant Professor of Industrial and
Systems Engineering

This dissertation was submitted to the Graduate Faculty of the Department of Decision and Information Sciences in the Warrington College of Business Administration and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August, 1999

Dean, Graduate School